

Pemrograman Dasar

# ***Phyton Web***

Menggunakan

## ***Framework Django***

untuk Pemula

Undang-Undang Republik Indonesia Nomor 28 Tahun 2014 tentang Hak Cipta

Pasal 1:

1. Hak Cipta adalah hak eksklusif pencipta yang timbul secara otomatis berdasarkan prinsip deklaratif setelah suatu ciptaan diwujudkan dalam bentuk nyata tanpa mengurangi pembatasan sesuai dengan ketentuan peraturan perundang undangan.

Pasal 9:

2. Pencipta atau Pengarang Hak Cipta sebagaimana dimaksud dalam pasal 8 memiliki hak ekonomi untuk melakukan a. Penerbitan Ciptaan; b. Penggandaan Ciptaan dalam segala bentuknya; c. Penerjemahan Ciptaan; d. Pengadaptasian, pengarangsemen, atau pentransformasian Ciptaan; e. Pendistribusian Ciptaan atau salinan; f. Pertunjukan Ciptaan; g. Pengumuman Ciptaan; h. Komunikasi Ciptaan; dan i. Penyewaan Ciptaan.

Sanksi Pelanggaran Pasal 113

1. Setiap orang yang dengan tanpa hak melakukan pelanggaran hak ekonomi sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf i untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 1 (satu) tahun dan/atau pidana denda paling banyak Rp100.000.000,00 (seratus juta rupiah).
2. Setiap Orang yang dengan tanpa hak dan/atau tanpa izin Pencipta atau pemegang Hak Cipta melakukan pelanggaran hak ekonomi Pencipta sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf c, huruf d, huruf f, dan/atau huruf h untuk Penggunaan Secara Komersial dipidana dengan pidana penjara paling lama 3 (tiga) tahun dan/atau pidana denda paling banyak Rp500.000.000,00 (lima ratus juta rupiah).

Rahma Yanti  
Fajri Rinaldi Chan

Pemrograman Dasar

# ***Phyton Web***

Menggunakan

***Framework Django***  
untuk Pemula



Penerbit Lakeisha  
2023

**PEMOGRAMAN DASAR *PHYTON WEB* MENGGUNAKAN  
*FRAMEWORK DJANGO* UNTUK PEMULA**

Penulis:

**Rahma Yanti**

**Fajri Rinaldi Chan**

Editor :

**Firdaus Annas**

Layout : Yusuf Deni Kristanto, S.Pd

Desain Cover : Tim Lakeisha

Cetak I September 2023

15,5 cm × 23 cm, 222 Halaman

ISBN: 978-623-420-913-6

Diterbitkan oleh Penerbit Lakeisha

**(Anggota IKAPI No.181/JTE/2019)**

Redaksi

Srikaton, RT 003, RW 001, Pucangmiliran, Tulung, Klaten, Jawa Tengah

Hp. 08989880852, Email: [penerbit\\_lakeisha@yahoo.com](mailto:penerbit_lakeisha@yahoo.com)

Website: [www.penerbitlakeisha.com](http://www.penerbitlakeisha.com)

Hak Cipta dilindungi Undang-Undang

Dilarang memperbanyak karya tulis ini dalam bentuk dan  
dengan cara apapun tanpa izin tertulis dari penerbit



## KATA PENGANTAR

**B**uku ini ditujukan untuk pemula yang ingin mempelajari dasar-dasar pemrograman dengan menggunakan bahasa pemrograman Python dan database MySQL. Buku ini akan memberikan pengantar yang komprehensif dan langkah-demi-langkah dalam mempelajari kedua teknologi ini.

Python adalah bahasa pemrograman yang populer dan mudah dipelajari. Buku ini akan memandu pembaca melalui konsep-konsep dasar dalam pemrograman menggunakan Python. Dimulai dari pengenalan tentang sintaks dasar, tipe data, operasi matematika, percabangan, perulangan, hingga penggunaan fungsi dan modul, pembaca akan memperoleh pemahaman yang kokoh tentang dasar-dasar pemrograman dengan Python.

Selain pemrograman Python, buku ini juga memperkenalkan database MySQL yang merupakan salah satu sistem manajemen basis data (DBMS) yang paling populer dan digunakan secara luas. Pembaca akan mempelajari cara menginstal MySQL, membuat database, mengelola tabel, dan melakukan manipulasi data menggunakan bahasa SQL.

Penulis

(            )



# PENDAHULUAN

**S**elamat datang di buku "Pemrograman Dasar Python Web Menggunakan Framework Django untuk Pemula." Buku ini merupakan panduan komprehensif yang dirancang khusus untuk membantu Anda memahami dasar-dasar pemrograman web menggunakan bahasa pemrograman Python dan framework Django. Baik Anda seorang pemula yang baru saja memasuki dunia pemrograman web atau seorang pengembang yang ingin menggali lebih dalam ke dalam Django, buku ini akan memberikan dasar yang kuat untuk memulai perjalanan Anda.

## **Mengapa Buku Ini Penting?**

Dalam era digital yang semakin berkembang, pemrograman web telah menjadi salah satu keterampilan yang paling dicari. Python, dengan sintaksis yang mudah dipahami, telah menjadi bahasa yang sangat populer untuk pengembangan web. Django, di sisi lain, adalah salah satu framework web Python yang paling kuat dan produktif. Menggabungkan keduanya adalah langkah cerdas untuk memahami dunia pemrograman web.

- Buku ini bertujuan untuk membantu Anda:
- Memahami dasar-dasar pemrograman web.
- Menguasai bahasa pemrograman Python.
- Membangun aplikasi web yang dinamis dengan menggunakan Django.
- Memahami konsep fundamental seperti model-view-controller (MVC) dalam pengembangan web.
- Memahami bagaimana Django mempermudah proses pengembangan web.

- Membangun proyek web pertama Anda dari awal hingga akhir.

### **Bagaimana Buku Ini Diatur?**

Buku ini dirancang dengan bahasa yang mudah dipahami, konsep yang terstruktur, dan banyak contoh kode. Setiap bab akan memandu Anda melalui langkah-langkah yang diperlukan untuk memahami dan menguasai topik tertentu. Selain itu, kami akan memberikan proyek nyata yang akan Anda bangun secara bertahap, sehingga Anda dapat melihat bagaimana konsep-konsep ini diterapkan dalam konteks praktis.

### **Apa yang Dibutuhkan?**

Untuk mendapatkan manfaat maksimal dari buku ini, Anda hanya perlu memiliki pengetahuan dasar tentang pemrograman. Bahkan jika Anda belum pernah mencoba Python atau Django sebelumnya, jangan khawatir; buku ini akan memandu Anda melalui langkah-langkah awal.

### **Siap Melangkah?**

Selamat, Anda telah mengambil langkah pertama menuju perjalanan yang menarik dalam dunia pemrograman web dengan Python dan Django. Segera, kita akan memulai dengan pemahaman dasar-dasar Python. Pastikan untuk mengikuti setiap bab dengan tekun, dan Anda akan segera merasakan kepercayaan diri Anda tumbuh sebagai seorang pengembang web.



# DAFTAR ISI

<b>KATA PENGANTAR.....</b>	<b>v</b>
<b>PENDAHULUAN .....</b>	<b>vi</b>
<b>DAFTAR ISI.....</b>	<b>viii</b>
<b>Daftar Gambar .....</b>	<b>xi</b>
<b>BAB I     <b>MENGENAL PHYTON.....</b></b>	<b>1</b>
Sekilas Tentang Phyton.....	1
Sejarah Python .....	3
<i>Tools</i> Yang di Butuhkan.....	3
Instal Server MySQL.....	3
XAMPP untuh Phyton .....	4
Editor Untuk Phyton .....	10
<b>BAB II    <b>DASAR DASAR WEBSITE .....</b></b>	<b>24</b>
Pengenalan Website .....	24
Pengenalan HTML .....	25
Struktur Dasar HTML.....	26
<i>Tag Tag</i> HTML.....	26
Elemen dalam HTML .....	27
<b>BAB III   <b>PEMROGRAMAN PYTHON .....</b></b>	<b>32</b>
Pengenalan Phython.....	32
Bahasa Pemrograman Phyton .....	33
Variabel dan Tipe Data dan Operator .....	33
Sintaks dasar Phython.....	40
Operasi Matematika .....	44
Struktur Kontrol : Percabangan dan Perulangan.....	57
Fungsi Pada Phyton.....	71
Kelas dan Objek.....	81
Penanganan Eksepsi .....	96



<b>BAB IV</b>	<b>HELLO DJANGO .....</b>	<b>98</b>
	Sejarah Django .....	98
	Pengertian Django .....	98
	Instalasi Django dan Setup Project.....	99
	Aplikasi Pertama Dengan Django .....	102
	URL dan Views.....	106
<b>BAB V</b>	<b>PROJECT DJANGO MEMBUAT BLOG PRIBADI.....</b>	<b>110</b>
	Inisiasi Project.....	110
	Membuat virtual environment .....	111
	Mengaktifkan virtual environment .....	112
	Instalasi dan membuat project Django .....	112
	Instalasi Library pendukung.....	114
	Membuat database.....	116
	Membuat app blog.....	117
	Menambahkan folder Template dan folder .....	120
	Membuat file base.html didalam folder templates/blog.....	121
	Membuat file header.html di dalam folder templates/blog.....	127
	Membuat file home.html di dalam folder templates/blog.....	129
	Membuat file blog_detail.html di dalam folder templates/blog.....	138
	Mengatur STATIC dan MEDIA ROOT Pada setting.py.....	139
	Memodifikasi file urls.py .....	140
	Memodifikasi file models.py di dalam folder blog.....	141
	Menambahkan file forms.py di dalam folder blog.....	144
	Memodifikasi file views.py di folder blog.....	144
	Membuat app Admin Blog .....	147
	Menambahkan 'admion_blog' difile settings.py pada variable INSTALLED_APPS .....	148
	Membahkan file folder templates dan static di app admin blog .....	149
	Membuat file base.html di dalam folder templates/auth di app admin_blog.....	149
	Membuat file login.html di dalam folder templates/auth di app admin_blog.....	151

Membuat file signup.html di dalam folder templates/auth di app admin_blog .....	154
Membuat file base.html di dalam folder templates/blog_admin di app admin_blog .....	156
Membuat file sidebar.html di dalam folder templates/blog_admin di app admin_blog .....	160
Membuat file artikel_add.html di dalam folder templates/blog_admin di app admin_blog .....	163
Membuat file artikel_list.html di dalam folder templates/blog_admin di app admin_blog .....	174
Membuat file artikel_update.html di dalam folder templates/blog_admin di app admin_blog .....	188
Membuat file artikel_category_list.html di dalam folder templates/blog_admin di app admin_blog .....	199
Memodifikasi file models.py di app admin_blog.....	210
Menambahkan file forms.py di dalam app admin_blog .....	211
Memodifikasi file views.py di app admin_blog.....	212
Memodifikasi file urls.py .....	218
Melakukan Migration ke project admin blog .....	218

**BAB VI PENUTUP ..... 221**

**DAFTAR PUSTAKA ..... 222**



## DAFTAR GAMBAR

Gambar 1 Halaman Download Aplikasi XAMPP .....	4
Gambar 2. Langkah Penginstalan Xampp Run administrator .....	5
Gambar 3. Langkah Penginstalan Xampp Setup .....	5
Gambar 4. Langkah Penginstalan Xampp Select Components .....	6
Gambar 5. Langkah Penginstalan Xampp Pilih Lokasi Penyimpanan.....	6
Gambar 6. Langkah Penginstalan Xampp Pilih Bahasa .....	7
Gambar 7. Langkah Penginstalan Xampp Ready to Install .....	7
Gambar 8. Proses Installing .....	8
Gambar 9. Allow Apache .....	8
Gambar 10. Finish Install .....	9
Gambar 11. Menjalankan Xampp .....	9
Gambar 12. Notepad++.....	10
Gambar 13. Visual Studio Code .....	12
Gambar 14. Sublime Text .....	14
Gambar 15. Halaman Download Visual Studio Code .....	15
Gambar 16. Install Visual Studio Code .....	16
Gambar 17. Penginstalan Visual Studio Code .....	16
Gambar 18. Pemilihan Install Tools Visual Studio Code .....	17
Gambar 19. Ready to Install Visual Studio Code .....	17
Gambar 20. Lembar Kerja Visual Studio Code .....	18

Gambar 21. Halaman Download Python.....	18
Gambar 22. Pemilihan Versi Download Python .....	19
Gambar 23. Memulai Instal Python .....	19
Gambar 24. Halaman Centang Opsi Python.....	20
Gambar 25.Instal Now Python .....	20
Gambar 26. Proses Installing .....	21
Gambar 27. Install Telah Selesai .....	21
Gambar 28. Halaman Visual Studio Code .....	22
Gambar 29. Ekstensi Python .....	22
Gambar 30. Penginstalan Ekstensi Python .....	23
Gambar 31. Hello World.....	41
Gambar 32. Fungsi Pada Python .....	73
Gambar 33. Parameter Operasional.....	75
Gambar 34. Lebih Dari Satu Return .....	78
Gambar 35. Ruang Lingkup Variabel dan Fungsi.....	79
Gambar 36. Output Ruang Lingkup Variabel dan Fungsi.....	80
Gambar 37. Atribut Pada Python .....	83
Gambar 38. Polimerfime Pada Python.....	88
Gambar 39. Fungsi Polimerfime Len () Pada Python.....	89
Gambar 40. Polimerfime Kelas Pada Python .....	90
Gambar 41. Ouput Polimerfime Kelas Pada Python .....	91
Gambar 42. Penggantian Metode.....	93
Gambar 43.Ouput Penggantian Metode .....	95
Gambar 44.Logika Polimorfime .....	95
Gambar 45. Membuat Virtual Environtment.....	100
Gambar 46. Output Virtual Environtment.....	101

Gambar 47. Powershall.....	102
Gambar 48. Memasukkan URL .....	102
Gambar 49. Menambahkan App Baru .....	103
Gambar 50. Membuat Website Hello World .....	104
Gambar 51. File Setting.....	105
Gambar 52. File urls.py .....	106
Gambar 53. Tampilan Hasil Website Hello World.....	106
Gambar 54. Class Diagram.....	110
Gambar 55. Windows Powershell.....	111
Gambar 56. Windows Powershell python -m menv env.....	111
Gambar 57. Windows Activate.....	112
Gambar 58. Pip Install Django.....	113
Gambar 59. Folder Django .....	113
Gambar 60. Blog WebApp .....	114
Gambar 61. Terminal My Library Sql Client .....	115
Gambar 62. Membuat Database.....	116
Gambar 63. Tab Database.....	117
Gambar 64. Mengisikan Nama Database.....	117
Gambar 65. Membuat App Blog .....	118
Gambar 66. Django Admin .....	118
Gambar 67. Setting.py.....	119
Gambar 68. Variable Database .....	119
Gambar 69. Password localhost.....	120
Gambar 70. Menambahkan Folder Template .....	121
Gambar 71. Halaman Blog Saya .....	147
Gambar 72. Halaman Content Blog.....	147

Gambar 73. App Admin Blog..... 148  
Gambar 74. Django-Admin ..... 148  
Gambar 75. File Folder Template ..... 149  
Gambar 76. Halaman Admin Blog ..... 219  
Gambar 77. Halaman Blog ..... 219  
Gambar 78. List Blog ..... 220  
Gambar 79. Category Blog ..... 220



# MENGENAL PHYTON

## Sekilas Tentang Phyton

Ada beberapa alasan mengapa menggunakan bahasa pemrograman Python:

**Mudah dipelajari dan digunakan:** Python dirancang dengan fokus pada kejelasan dan keterbacaan kode, sehingga mudah dipelajari, terutama bagi pemula dalam pemrograman. Syntax yang sederhana dan struktur yang intuitif membuatnya mudah dipahami dan digunakan.

**Kaya dengan pustaka dan modul:** Python memiliki ekosistem yang sangat luas dengan banyak pustaka dan modul yang telah dikembangkan oleh komunitas yang besar. Pustaka-pustaka ini memudahkan dalam pengembangan berbagai jenis aplikasi, termasuk pengolahan data, kecerdasan buatan, pengembangan web, analisis statistik, dan banyak lagi. Dengan menggunakan pustaka yang sudah ada, pengembang dapat menghemat waktu dan usaha dalam mengimplementasikan fitur-fitur kompleks.

**Portabilitas dan kompatibilitas:** Python dapat berjalan di berbagai sistem operasi utama, termasuk Windows, macOS, Linux, dan juga di lingkungan yang berbeda seperti server web dan

platform mobile. Ini membuatnya sangat portabel dan mudah untuk digunakan dalam berbagai lingkungan.

**Produktivitas tinggi:** Python adalah bahasa pemrograman tingkat tinggi yang menyediakan banyak fitur dan abstraksi yang memungkinkan pengembang untuk menulis kode dengan lebih sedikit baris dibandingkan dengan bahasa pemrograman lainnya. Ini berarti pengembang dapat mencapai lebih banyak dengan usaha yang lebih sedikit dan mengurangi waktu pengembangan.

**Komunitas yang besar:** Python memiliki komunitas pengguna yang besar dan aktif di seluruh dunia. Komunitas ini menyediakan dukungan yang luas dalam bentuk dokumentasi, forum diskusi, dan berbagi pengetahuan. Jadi, jika Anda mengalami masalah atau membutuhkan bantuan, kemungkinan besar akan ada seseorang yang dapat membantu Anda.

**Dukungan untuk pemrograman lanjutan:** Python mendukung konsep-konsep pemrograman lanjutan seperti pemrograman berorientasi objek, pemrograman fungsional, penanganan kesalahan, dan banyak lagi. Ini memungkinkan pengembang untuk mengembangkan aplikasi yang lebih kompleks dan dapat dengan mudah memperluas kemampuan Python sesuai kebutuhan.

**Popularitas dan pertumbuhan:** Python telah menjadi salah satu bahasa pemrograman yang paling populer di dunia. Keberhasilannya terutama terletak pada kegunaannya dalam berbagai bidang seperti pengolahan data, kecerdasan buatan, pengembangan web, dan sebagainya. Popularitas yang tinggi ini berarti ada banyak sumber daya dan komunitas yang tersedia, serta banyak kesempatan kerja untuk pengembang Python.

Tentu saja, pemilihan bahasa pemrograman tergantung pada kebutuhan dan preferensi individu, tetapi Python menawarkan banyak keunggulan yang menjadikannya pilihan yang populer dan kuat untuk berbagai jenis pengembangan perangkat lunak.



## Sejarah Python

Sejarah python dikembangkan oleh Guido van Rossum pada tahun 1990 di CWI, Amsterdam sebagai kelanjutan dari Bahasa pemrograman ABC. Versi terakhir yang dikeluarkan CWI adalah 1.2. tahun 1995, Guido pindah ke CNRI sambal terus melanjutkan pengembangan Python. Python adalah Bahasa pemrograman interpretatif multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. Python diklaim sebagai Bahasa yang menggabungkan kapabilitas, kemampuan, dengan sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas Pustaka standar yang besar serta komprehensif. Python mendukung multi paradigma pemrograman, utamanya, namun tidak dibatasi. Salah satu fitur yang tersedia pada python adalah sebagai Bahasa pemrograman dinamis yang dilengkapi dengan manajemen memori otomatis. Seperti halnya pada Bahasa pemrograman dinamis lainnya, python umumnya digunakan sebagai Bahasa skrip meski pada praktiknya penggunaan Bahasa ini lebih luas mencakup konteks pemanfaatan yang umumnya tidak dilakukan menggunakan Bahasa skrip. Python dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai platform system.

## Tools Yang di Butuhkan

Untuk bisa menguji dan menjalankan kode Python, kamu dapat mengunduh aplikasi seperti XAMPP yang digunakan untuk mengubah computer pribadi menjadi server. Sedangkan untuk menggunakan MySQL, kamu juga bisa mengunduh aplikasi MySQL secara gratis dari situs resminya sendiri.

## Instal Server MySQL

Komputer tidak dapat mengolah kode-kode Python tanpa memanfaatkan tool tambahan. Agar kamu bisa menguji script atau kode-kode Python di computer sendiri, langkah pertama yang

dapat dilakukan yaitu kita menginstalasi server yang sanggup memproses Phyton. Ada beberapa perangkat lunak yang bisa digunakan. Akan tetapi kami merekomendasikan yaitu XAMPP.

## XAMPP untuk Phyton

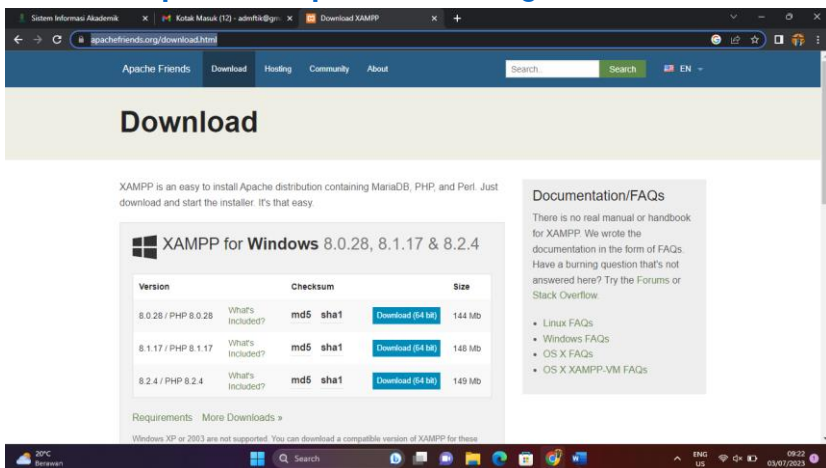
Xampp merupakan server yang paling familiar digunakan untuk keperluan belajar Bahasa Phyton secara mandiri, terutama bagi programmer pemula. Selain free, fitur pun tergolong lengkap dan gampang digunakan oleh programmer Phyton tingkat awal, yang perlu dilakukan hanyalah menjalankan module apache yang ada di dalam XAMPP tersebut.

Ada beberapa pilihan versi XAMPP. Unduh versi xampp yang terbaru yaitu versi 8.2.4

### Langkah Instalasi XAMPP

1. Langkah pertama yang harus kita lakukan yaitu mendownload xampp :

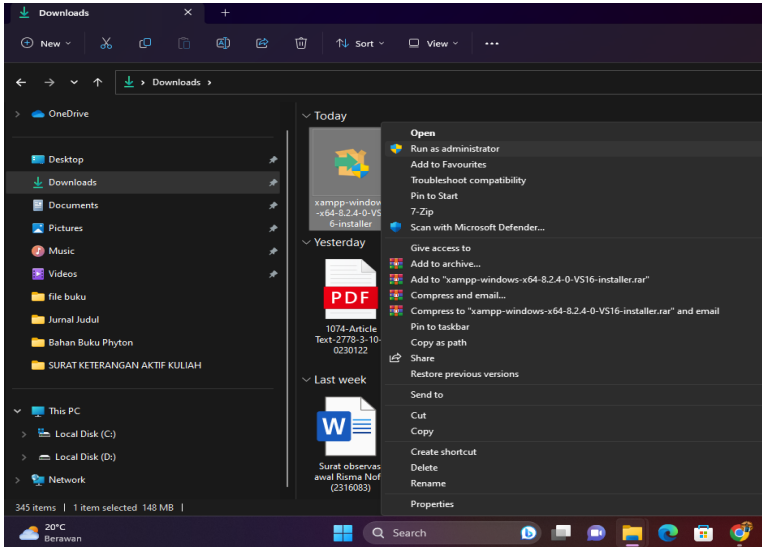
**Sumber :** <https://www.apachefriends.org/download.html>



**Gambar 1** Halaman Download Aplikasi XAMPP

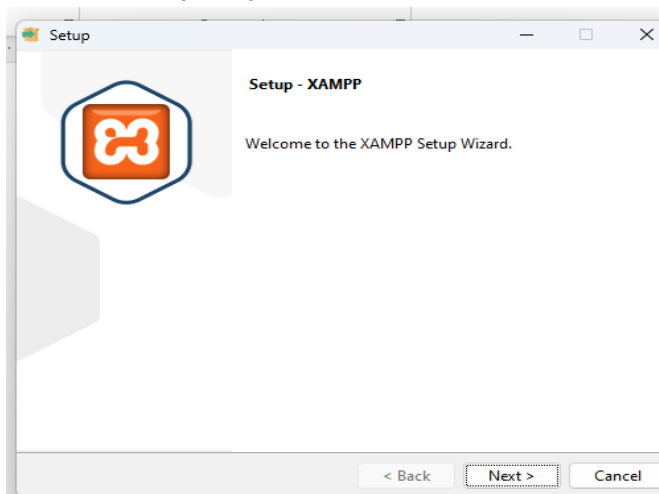
Kita pilih yang paling bawah lalu klik download.

2. Setelah selesai mendownload , kita bisa mulai untuk menginstal XAMPP, kita buka dalam folder download ada master XAMPP yang sudah terdownload lalu klik kanan pilih >run as administrator.



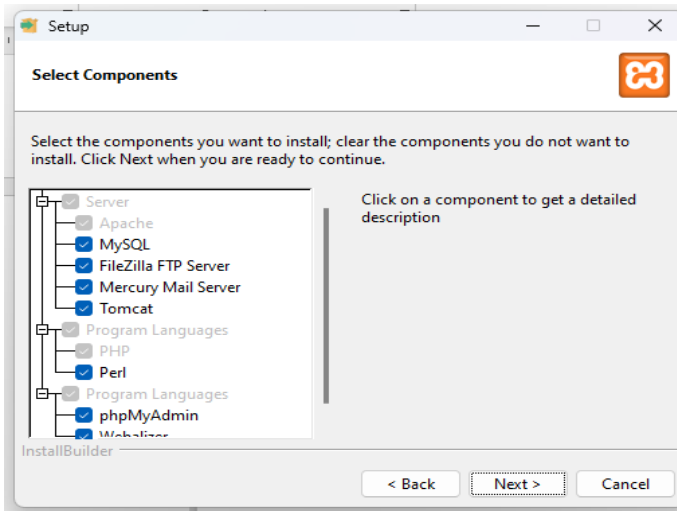
**Gambar 2. Langkah Penginstalan Xampp Run administrator**

3. Untuk langkah selanjutnya kita akan klik next



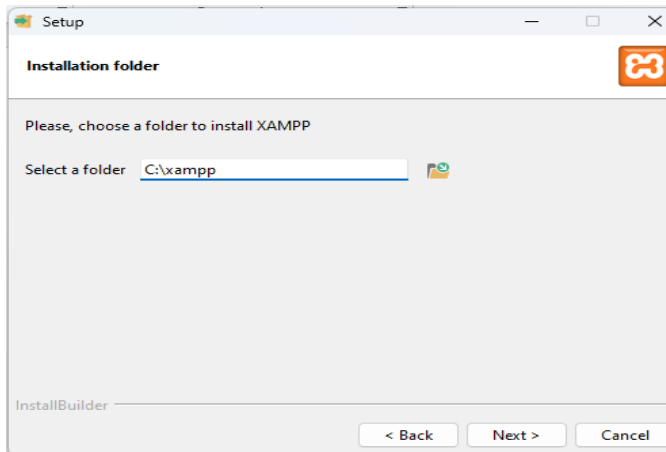
**Gambar 3. Langkah Penginstalan Xampp Setup**

4. Kemudian dilangka ini kita klik **next**



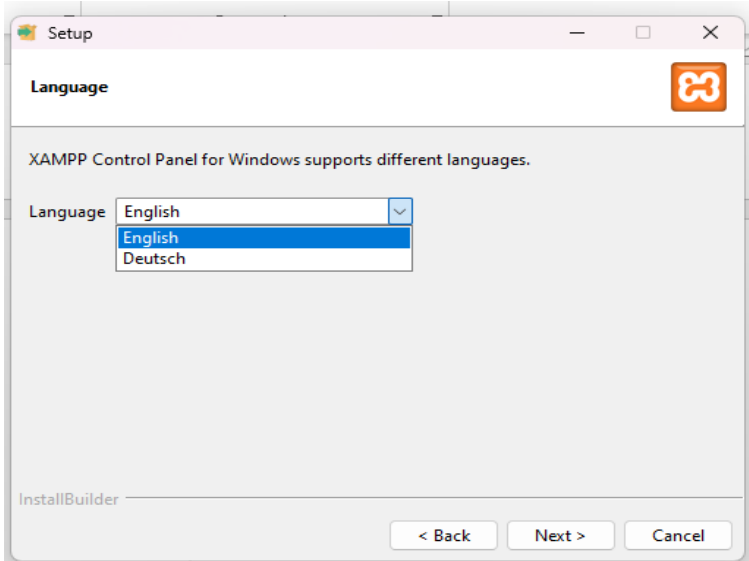
**Gambar 4. Langkah Penginstalan Xampp Select Components**

5. Untuk pemilihan folder penyimpanan kita pilih misalnya saya memilihnya di Localdisk :C , kemudian setelah dipilih kita klik **next**.



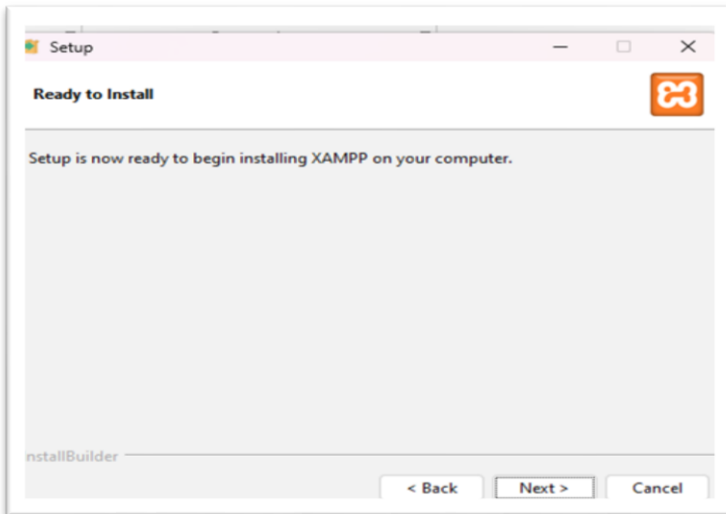
**Gambar 5. Langkah Penginstalan Xampp Pilih Lokasi Penyimpanan**

6. Kemudian pada pemilihan Bahasa kita pilih Bahasa English



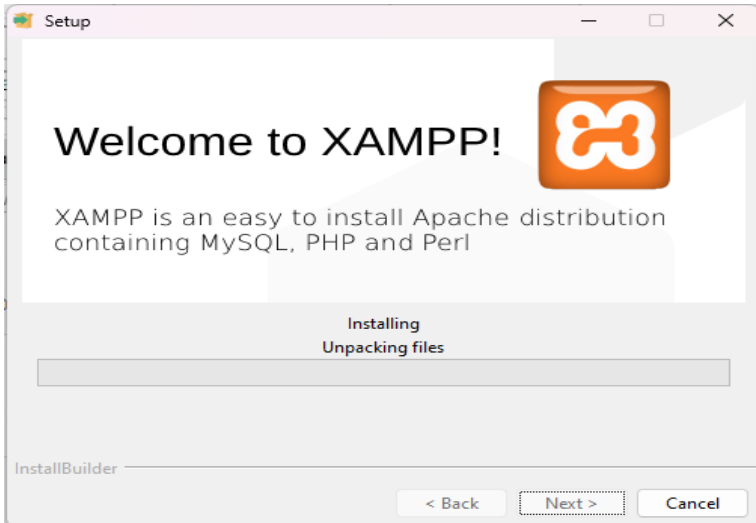
**Gambar 6. Langkah Penginstalan Xampp Pilih Bahasa**

7. Pada langkah ini kita klik next



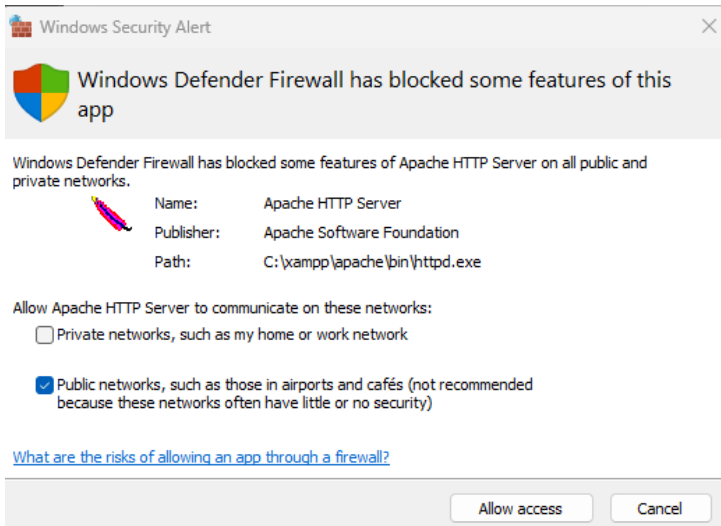
**Gambar 7. Langkah Penginstalan Xampp Ready to Install**

8. Selanjutnya kita menunggu proses installing dari XAMPP



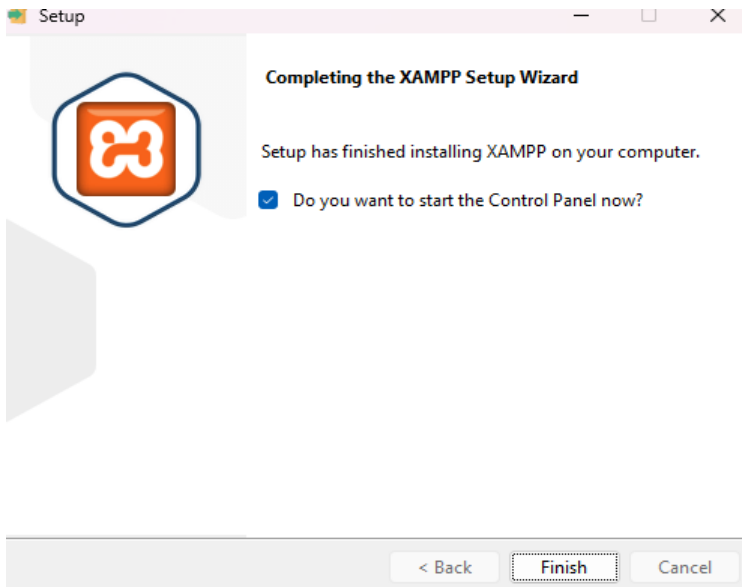
**Gambar 8. Proses Installing**

9. Setelah selesai proses maka akan muncul tampilan seperti dibawah ini kita klik **allow**



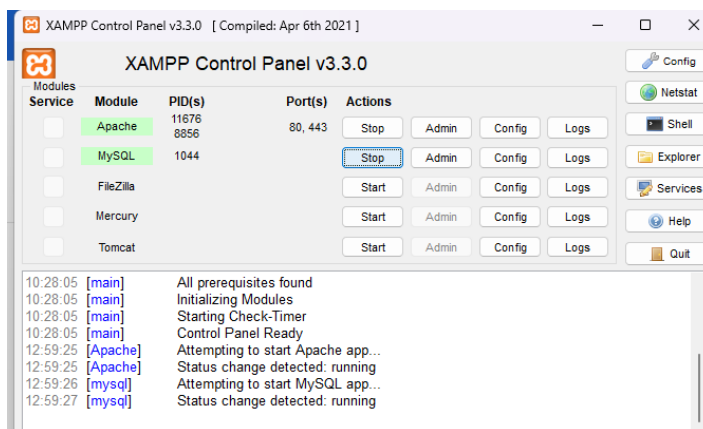
**Gambar 9. Allow Apache**

10. Jika sudah selesai maka kita klik **finish**



**Gambar 10. Finish Install**

Setelah itu kita star kan **apache dan mysql**



**Gambar 11. Menjalankan Xampp**

## Editor Untuk Python

Ada beberapa editor untuk Bahasa Python diantaranya:

### a) Notepad++



**Gambar 12. Notepad++**

Notepad++ adalah editor teks dan editor kode sumber yang berjalan pada sistem operasi windows. Notepad++ menggunakan komponen Scintilla untuk melihat dan mengedit file teks dan kode sumber kedalam berbagai bahasa pemrograman. Notepad++ didistribusikan sebagai perangkat lunak gratis. Proyek ini dicetuskan oleh Sourceforge.net, memiliki lebih dari 27 juta unduhan dan telah memenangkan penghargaan pilihan komunikasi SourceForge Community Choice Award for Best Developer Tool. Pengembang dari Notepad++ adalah Don Ho, dan merilis aplikasi ini pada tanggal 24 November 2003. Software ini telah memiliki lisensi dari GNU (General Public License) dengan ukuran program yang kecil yaitu 5.5mb. Bahasa pemrograman yang didukung oleh Notepad++ adalah Bahasa C++ karena fungsi-fungsinya yang dimasukkan kedalam daftar fungsi dan kata-kata akan berubah sesuai dengan makna kata C++.



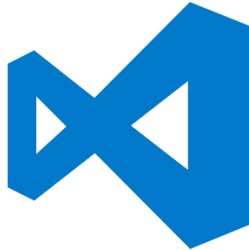
### **Kelebihan :**

- 1) Simple, Ringan dan Cepat dibandingkan dengan text editor lainnya, notepad++ tidak perlu menunggu loading opening library, terutama untuk PC/laptop low-spec, seperti software Adobe Dreamweaver dan Eclipse.
- 2) Bracket Matching atau bisa dibilang mengumpulkan yang sesuai, biasanya digunakan pada saat menuliskan syntax percabang, perulang dan bagian utama program. Fungsi ini berguna ketika menuliskan block program yang panjang dan terkadang kita lupa untuk menutup block program yang telah kita buat.
- 3) Syntax Highlighting tamping source code, disini kita dapat melihat warna pada setiap fungsi dari syntax. Sehingga tidak bingung menggolongkan kegunaan syntax yang kita tulis dan dapat dibaca dengan mudah. Contoh tulis yang berwarna hijau biasanya terdapat pada statement jika kita menuliskan komentar pada sebuah program.
- 4) Syntax Folding atau dikenal dengan melipat source code. Hal ini hampir sama seperti bracket matching yang dibahas sebelumnya. Bracket matching memiliki fungsi untuk menunjukkan suatu awal dan akhir dari block program. Sedangkan, untuk Syntax Folding disini digunakan untuk menyembunyikan block program tertentu agar terlihat lebih ringkas tampilan agar programmer tidak perlu melihat seluruh syntaxnya apalagi jika sudah sampai lebih 1000 baris lebih.
- 5) Quick Color Picker++. Fungsi yang satu ini berguna pada saat kita menuliskan kode warna pada html ataupun CSS, kita tidak perlu menuliskan kode terlebih dahulu karena biasanya akan muncul kotak dengan aneka warna yang bisa langsung dipilih. Setelah memilih kotak warna, maka kode warna tersebut akan otomatis muncul.

**Kekurangan :**

- 1) Karakter dalam notepad++ sangat terbatas.
- 2) Format lebih sedikit.
- 3) Pengolahan kata dasar.

**b) Visual Studio Code**



**Gambar 13. Visual Studio Code**

*Visual Studio Code* adalah aplikasi *editor* teks gratis di kembangkan oleh *Microsoft* yang dapat digunakan di semua bahasa pemrograman yang ada tanpa perlu berganti aplikasi *editor*, serta dapat dijalankan di berbagai *platform Operating System (OS)* seperti *windows,linux, dan mac OS*. *Visual Studio Code* memudahkan para *Programmer* saat berganti bahasa pemrograman tanpa perlu berganti aplikasi *editor* serta memahami dan konfigurasi *tools* Kembali di aplikasi *editor* barunya.*Visual Studio Code* juga memberikan kebebasan kepada penggunanya dalam tema, *debugger ,extension*, dan lainnya.

**Kelebihan :**

- 1) Dapat digunakan secara gratis  
Jika kamu masih awam dan menghemat biaya dalam pengembangan aplikasi, maka Visual Studio Code adalah solusinya. Meskipun memiliki berbagai kelebihan dan segudang fitur, namun kamu tetap bisa menggunakan Visual Studio Code secara gratis.

- 2) **Fitur yang lengkap**  
Menawarkan fitur yang lengkap menjadi kelebihan Visual Code Studio. Seperti yang sudah kita bahas bahwa fitur Visual Code Studio sangat lengkap dan bahkan ada fitur yang tidak dimiliki oleh pesaingnya yaitu Extension Marketplace. Dengan Extension Marketplace kamu bisa menambahkan berbagai tools secara bebas.
- 3) **Performa yang cepat**  
Performa menjadi kelebihan Visual Code Studio juga. Pasti jika kamu menggunakan aplikasi maka kamu akan menilai aplikasi tersebut berdasarkan performanya kan! Nah Visual Studio Code adalah code editor yang memiliki performa kilat karena ukurannya yang ringan. Meskipun kamu sudah mengunduh dan memasang banyak extension, namun visual studio code tetap mampu berjalan secara optimal. Alasannya karena code editor ini sudah dioptimasi sedemikian rupa agar tetap ringan dan banyaknya extension di dalamnya tidak akan terlalu mempengaruhi kinerja aplikasi ini.
- 4) **Mendukung banyak bahasa pemrograman**  
Visual studio code adalah code editor yang mendukung berbagai bahasa pemrograman. Kelebihan Visual Code Studio ini menjadikannya menjadi code editor yang cukup diminati karena memfasilitasi dalam pengembangan aplikasi. Salah satu contohnya adalah Electron yang berguna untuk Node.js dan JavaScript dan juga Monaco Cloud Editor yang mendukung HTML, .NET, Roslyn, dan lain-lain.
- 5) **Multiplatform**  
Meskipun dikembangkan oleh Microsoft, namun kelebihan Visual Code Studio yaitu mampu berjalan di berbagai sistem operasi seperti Mac Os, Windows, hingga Linux. Jadi kamu bisa menggunakan platform apa saja tanpa terkendala.

### **Kekurangan :**

Kekurangan vs code terletak pada performa karena dibandingkan dengan text editor lain seperti sublime text yang masih lancar di cpu 2 core dengan 2gb ram sedangkan dengan spek yang sama vs code kadang suka crash atau lag. Jadi untuk performa yang stabil vs code butuh cpu 4 core dan 4gb ram.

### **c) Sublime Text**



**Gambar 14. Sublime Text**

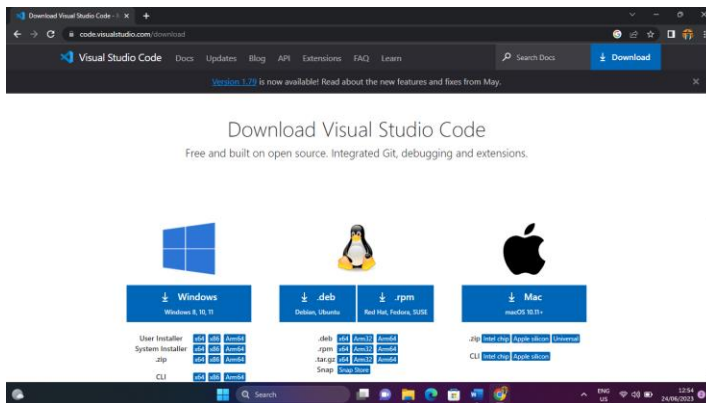
Sublime Text adalah sebuah aplikasi text editor untuk menulis kode program dan teks yang dapat berjalan diberbagai platform operating system dengan menggunakan teknologi Phyton API yang sudah diperbarui menjadi python 3.8. Sublime dibuat karena terinspirasi oleh Vim, Aplikasi ini sangat fleksibel dan powerfull. Fungsionalitas dari aplikasi ini dikembangkan dengan menggunakan sublime-packages. Sublime Text bukanlah aplikasi open source, aplikasi ini juga dapat digunakan free mode, akan tetapi beberapa fitur yang dikembangkan dari aplikasi ini merupakan hasil temuan dan mendapatkan dukungan penuh dari komunitas serta memiliki lisensi aplikasi gratis. Seblime Text merupakan *software* native support dan mendukung banyak programming languages yang bermacam-macam dan menyediakan fungsionalitas sintaks hampir di semua bahasa pemrograman dikembangkan, oleh komunitas programming languages seperti C, C++, C#, CSS, D, DYLAN, Erlang, HTML, Groovy, Haskell, Java, Javascript, LaTeX,

Lisp, Lua, Markdown, MATLAB, Ocaml, Perl, PHP, Python, R, Ruby, SQL, TCL, Textmate and XML. Biasanya bahasa pemrograman yang didukung ataupun belum didukung secara default dapat lebih dimaksimalkan atau didukung dengan menggunakan add-ons yang bisa didownload sesuai kebutuhan pengguna. Text editor ini pertama kali dirilis pada tanggal 18 Januari 2008, setelah 5 tahun kemudian versi kedua rilis dari text editor ini, untuk rilis stabil Sublime Text 3 pada tahun 2017. Namun kini versi Sublime Text Editor telah mencapai versi 4 yang dirilis pertama kali pada tahun 2021. Sublime Text didukung oleh beberapa sistem operasi seperti Linux, Mac OS, X dan Windows. Ada banyak fitur yang tersedia di Sublime Text seperti minimaps, side-by-side scripts dan bracketed highlight dan lain sebagainya.

## A. Instalasi Python

Sebelum kita menginstall python kita akan menginstall visual studio code terlebih dahulu, Adapun langkah dalam penginstalnya adalah sebagai berikut :

1) Mendownload software visual studio code

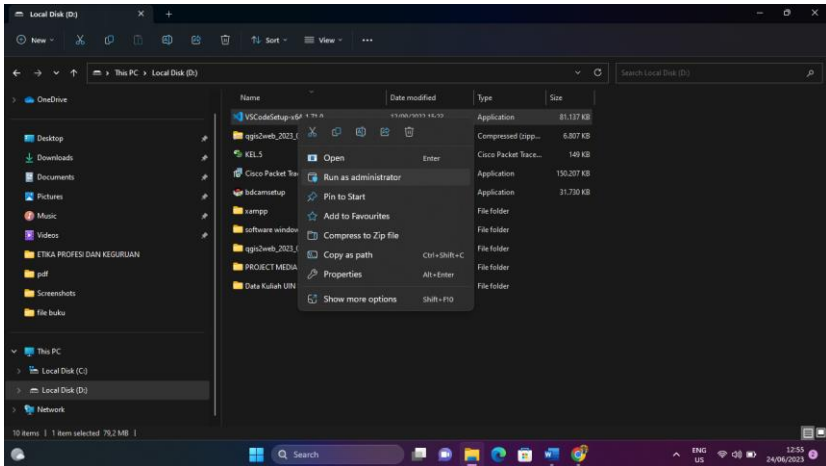


**Gambar 15. Halaman Download Visual Studio Code**

Sumber : <https://code.visualstudio.com/download>

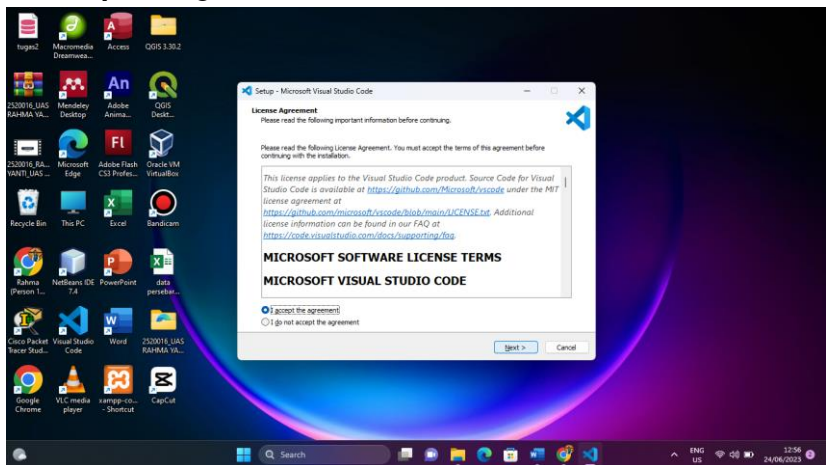
Pada Langkah pertama kita akan mendownload software sesuai dengan kebutuhan kita, misalnya kita memakai windows maka download yang windows.

- 2) Setelah didownload Langkah selanjutnya yaitu kita buka dimana tempat file tersebut, lalu klik kanan pilih run administrator.



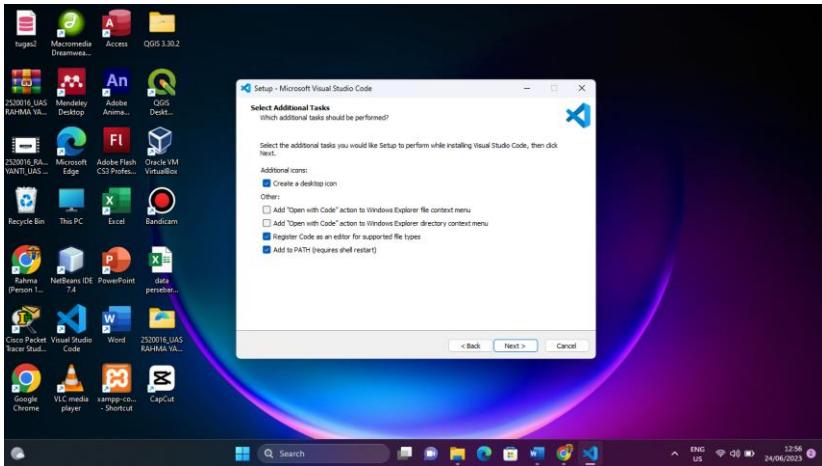
**Gambar 16. Install Visual Studio Code**

- 3) Lalu akan tampil tampilan seperti dibawah ini kita pilih yang **accept fot agreement>next**



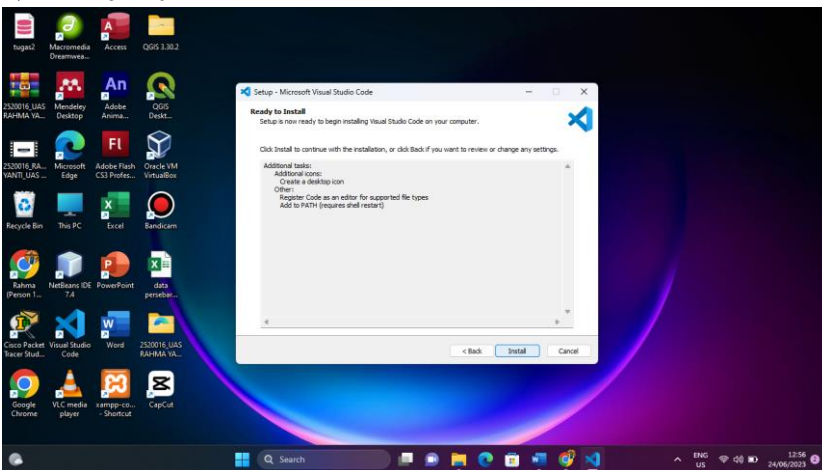
**Gambar 17. Penginstalan Visual Studio Code**

4) Pada tahap selanjutnya kita pilih **next**.



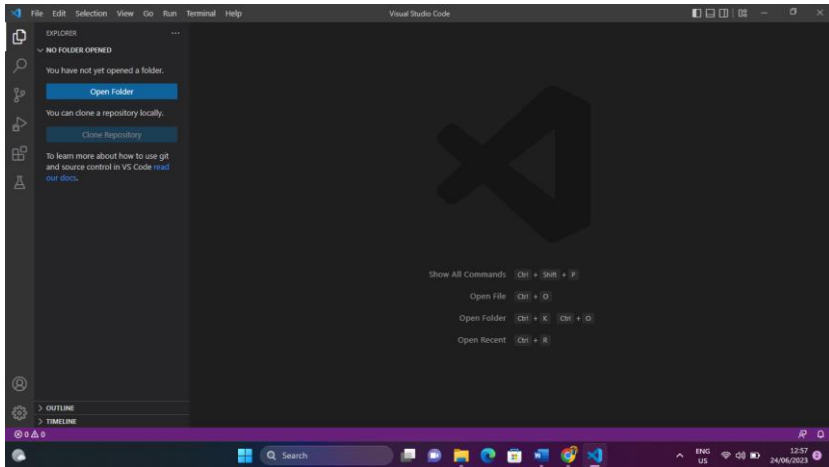
**Gambar 18. Pemilihan Install Tools Visual Studio Code**

5) Selanjutnya kita klik **Install**.



**Gambar 19. Ready to Install Visual Studio Code**

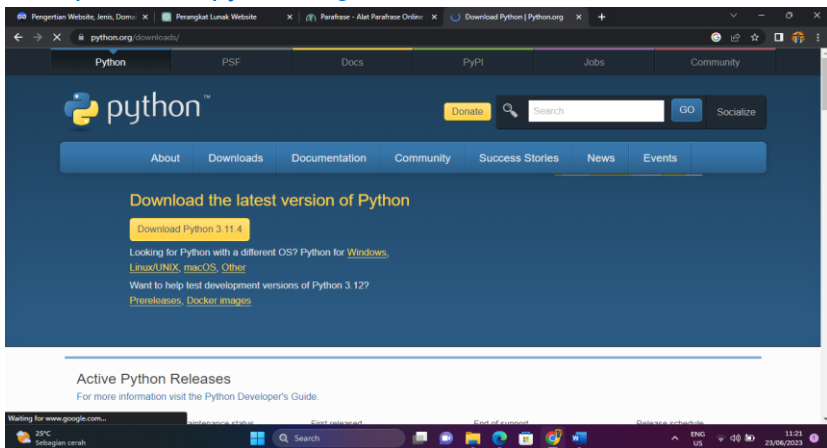
6) Jika telah selesai melakukan penginstalan maka akan tampil tampilan seperti gambar dibawah ini.



**Gambar 20. Lembar Kerja Visual Studio Code**

Setelah selesai menginstal visual studio code maka selanjutnya kita Menginstal Phyton, Adapun tahapan dalam penginstal phyton 3.11.4 yaitu :

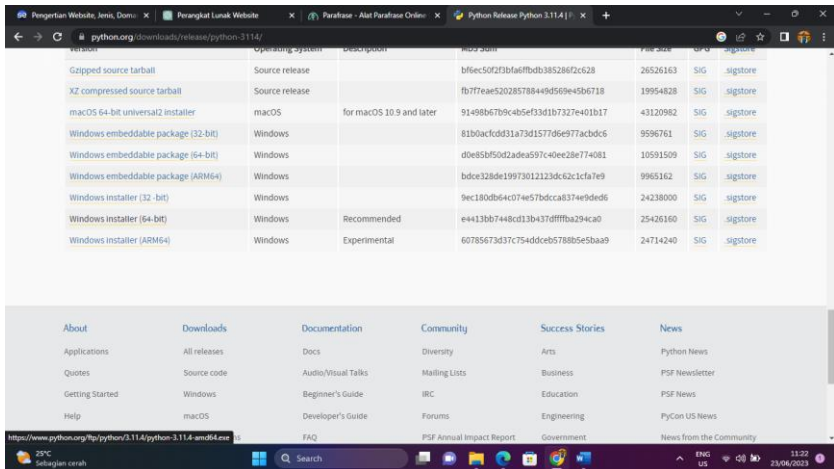
- 1) Mendownload software phython melalui website resmi python <https://www.python.org/downloads/>



**Gambar 21. Halaman Download Python**

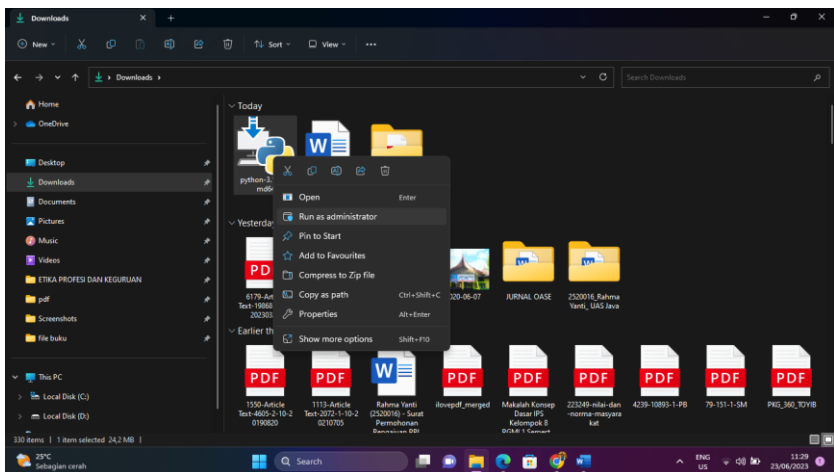


2) Pada saat mendownload kita pilih sesuai laptop kita jika laptop 64 bit maka pilih yang versi 64 bit seperti gambar dibawah ini.



Gambar 22. Pemilihan Versi Download Python

3) Jika sudah terdownload maka kita klik kanan lalu pilih run administrator



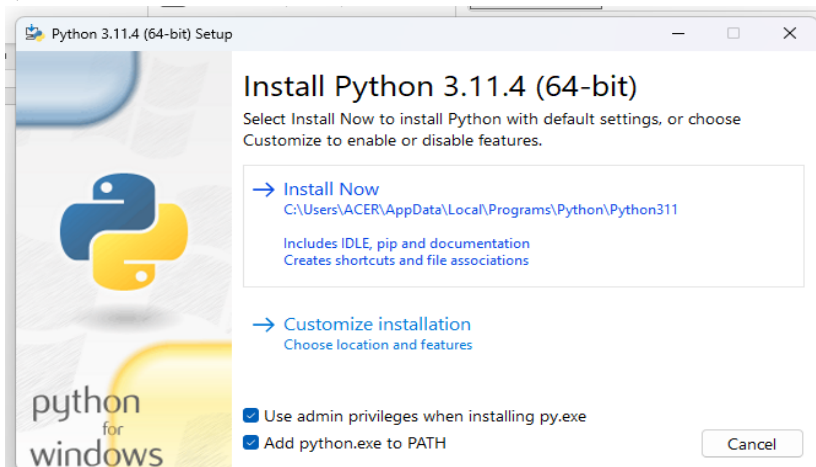
Gambar 23. Memulai Instal Python

4) Pada tahap selanjutnya kita centang keduanya.



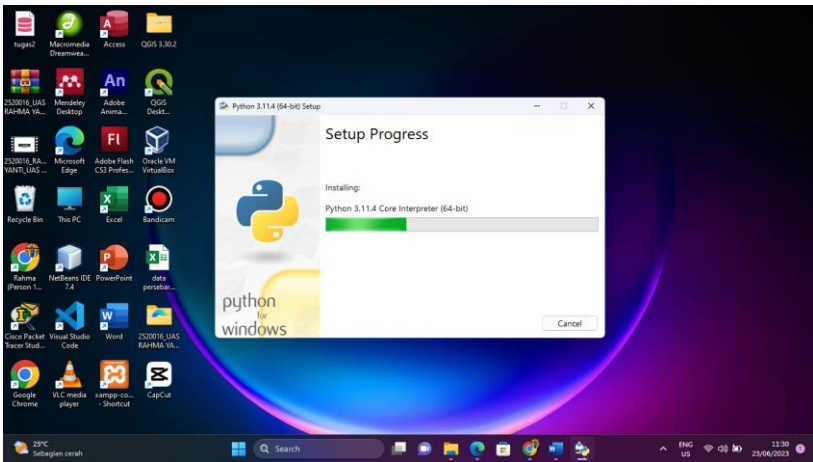
**Gambar 24. Halaman Centang Opsi Python**

5) Selanjutnya kita pilih install now



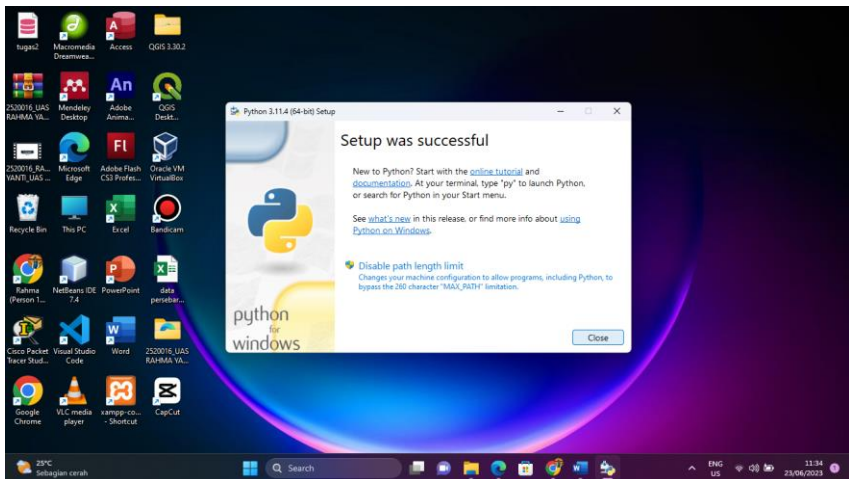
**Gambar 25. Instal Now Python**

6) Kita tunggu proses penginstalan untuk beberapa waktu



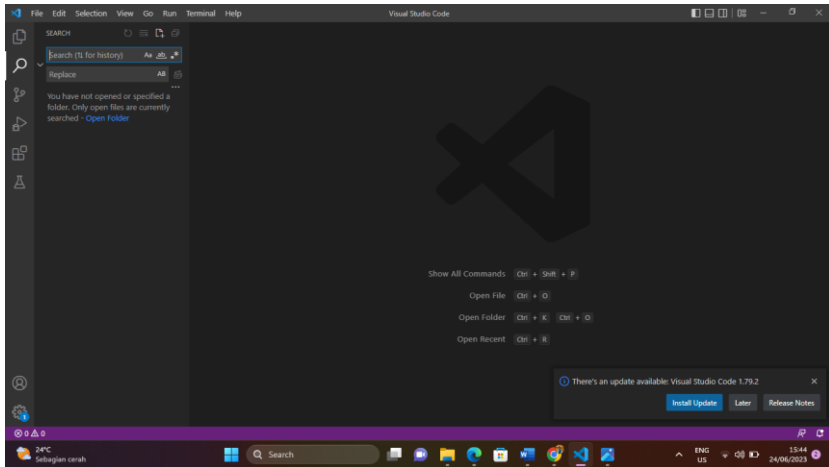
**Gambar 26. Proses Installing**

7) Jika sudah tampil seperti dibawah ini penginstalan telah selesai lalu kita klik close.



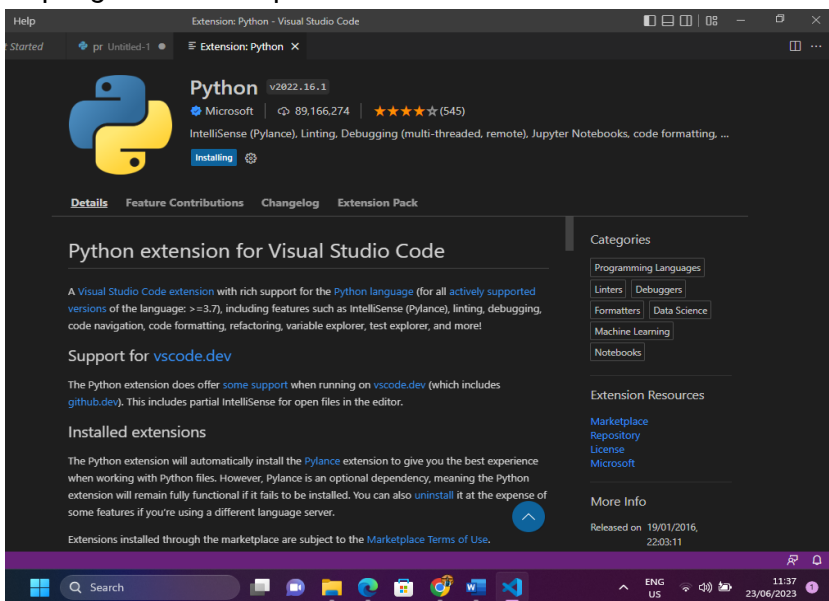
**Gambar 27. Install Telah Selesai**

8) Setelah itu kita buka aplikasi visual studio code yang telah kita instal diawal tadi lalu cari ekstensi python



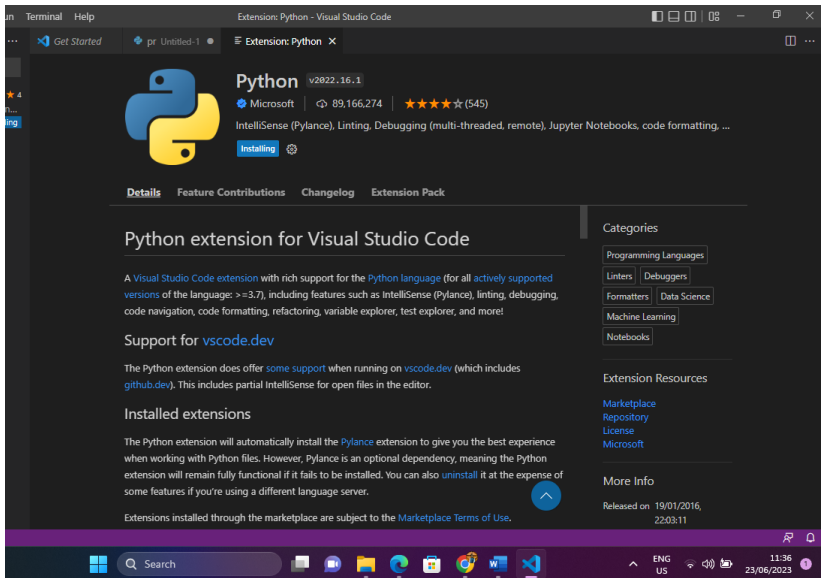
**Gambar 28. Halaman Visual Studio Code**

9) Jika sudah muncul maka kita klik instal dan tunggu proses penginstalan sampai selesai.



**Gambar 29. Ekstensi Python**

10) Jika sudah selesai ekstensi python sudah bisa digunakan.



**Gambar 30. Penginstalan Ekstensi Python**



## BAB II

# DASAR DASAR WEBSITE

## Pengenalan Website

Menurut Bekti website adalah kumpulan halaman-halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau gerak, animasi, suara, dan atau gabungan dari semuanya, baik yang bersifat statis maupun dinamis yang membentuk satu rangkaian konstruksi yang saling terkait, yang masing-masing dihubungkan dengan jaringan-jaringan halaman.

Halaman merupakan unsur utama dan paling mendasar dalam website. Setiap halaman akan berisi berbagai informasi yang ingin disampaikan dalam format teks, gambar, animasi, atau media lainnya, sesuai dengan kebutuhan. Halaman pada website akan dibatasi isinya agar fokus hanya membahas sesuatu yang dibutuhkan oleh pengguna. Website juga akan menjadi suatu konstruksi kesatuan yang memiliki navigasi untuk berpindah antarhalaman, agar memudahkan pengguna dalam memilah dan memilih konten yang diinginkan.

Sementara itu, Ginanjar (2014) berpendapat bahwa website adalah rangkaian atau sejumlah halaman web di internet yang memiliki topik saling berkaitan untuk mempresentasikan suatu

informasi. Jika website merupakan representasi dari suatu entitas seperti perusahaan, maka tentunya setiap topik informasi yang berada di dalamnya akan saling berkaitan, misalnya terdapat halaman: visi perusahaan, misi perusahaan, dan bisnis perlahan. Hal ini semakin mengonfirmasi bahwa website merupakan kesatuan konstruksi yang memiliki halaman-halaman terkait.

Selanjutnya, menurut Rahmadi (2013, hlm. 1) website atau lebih dikenal dengan sebutan situs adalah sejumlah halaman web yang memiliki topik saling terkait, terkadang disertai pula dengan berkas-berkas gambar, video atau jenis-jenis berkas lainnya.

Pengertian di atas tersebut mengingatkan kembali pada bermacam isi yang dapat dimuat oleh suatu website. Bagaimana jika website tidak hanya berisi informasi berupa teks saja, misalnya Youtube. Website youtube hampir tidak memiliki informasi biasa dan justru hanya menyajikan video yang diupload oleh penggunanya. Video-video yang dimunculkan juga terus berubah setiap saat dan seakan tidak memiliki informasi yang tetap.

## Pengenalan HTML

HTML bertanggung jawab mengatur tampilan halaman sebuah website. Meskipun mengandung beragam kode, tetapi HTML tidak bisa disebut sebuah Bahasa pemrograman. Mengapa? Bahasa pemrograman pada umumnya memiliki struktur tertentu, seperti ada logika if, pengulangan, variable, debugger, dan juga pada penulisan syntax yang benar-benar cermat. Struktur tersebut tidak ditemukan dalam HTML. Dalam HTML kesalahan pada penulisan kode tidak akan menghasilkan pesan kesalahan. Sebutan lazim untuk HTML adalah Bahasa markup (markup language) seperti yang ada didalam singkatan HTML itu sendiri. Itu artinya, HTML adalah Bahasa struktur untuk menandai bagian-bagian dari sebuah halaman.

## Struktur Dasar HTML

HTML (*Hypertext Markup Language*) memiliki struktur dasar yang terdiri dari :

- ✚ Tag DTD atau DOCTYPE
- ✚ Tag HTML
- ✚ Tag HEAD
- ✚ Tag BODY

Struktur di atas merupakan struktur HTML paling dasar. Lihat script berikut :

```
<!DOCTYPE html>
<html>
  <head>
    <title>Belajar HTML</title>
  </head>
  <body>
    <p> Hello Dunia !</p>
  </body>
</html>
```

## Tag Tag HTML

### Tag Doctype

Tag paling awal dari struktur dokumen HTML adalah *doctype* atau biasa disingkat DTD. DTD sendiri merupakan singkatan dari *Document Type Declaration* yang berfungsi untuk memberitahu browser bahwa dokumen yang akan ditampilkan adalah dokumen berjenis HTML.

*Doctype* memiliki berbagai versi cara penulisannya yang umumnya tergantung dari versi HTML yang digunakan. Dalam beberapa kasus, *doctype* bisa diabaikan. Jika tidak menulis *doctype* diawal struktur HTML, browser akan tetap memproses dan juga menampilkan halaman website kita seperti biasa. Namun, browser hanya akan menampilkan halaman HTML tersebut pada mode khusus yang disebut Quirk Mode. Itu artinya, browser menampilkan halaman web tidak sama persis seperti yang kita



inginkan karena dianggap halaman tersebut kemungkinan besar merupakan halaman web obsolete.

## **Tag HTML**

Setelah menulis tag *Doctype*, tag berikutnya yang harus dibuat adalah tag `<html>`. Tag ini merupakan tag pembuka dari seluruh halaman web yang akan kita buat. Selanjutnya semua kode HTML yang membentuk desain sebuah halaman website harus berada didalam tag `<html>` ini.

Tag `<html>` harus ditulis berpasangan, dimulai dengan `<html>` dan diakhiri dengan `</html>`, sementara kode-kode HTML lain berada diantara tag pembuka dan penutup tersebut.

## **Tag Head**

Tag yang ditulis dengan bentuk `<head>` ini berfungsi secara teknis. Bagian atau elemen yang ditulis pada tag `<head>` umumnya berbentuk keterangan teknis, berupa definisi dan judul halaman, kode-kode CSS, JavaScript, deskripsi halaman, dan kode lainnya terlihat sebagai salah satu desain didalam sebuah halaman.

Judul halaman website ditulis menggunakan tag `<title>` sesudah kita menulis tag `<head>`. Title ini biasanya ditampilkan pada bagian paling atas jendela browser atau dibagian tab (pada chrome).

## **Tag Body**

tag `<body>` digunakan untuk meletakkan semua elemen yang akan terlihat didalam halaman website pada saat halaman itu diakses oleh browser. Sama juga seperti

## **Elemen dalam HTML**

Ada beberapa elemen dalam HTML yaitu :

## a) HEAD

Head berfungsi untuk mengidentifikasi header web yang berisikan informasi tentang web. Bentuk umum tag head sebagai berikut:

```
<html>
<head> ... </head>
</html>
```

Tag HEAD memiliki beberapa atribut, antara lain: *title*, *meta*, *style*, *link*, *script*.

- **Tag TITLE**

Tag TITLE, digunakan untuk memberikan judul dari suatu homepage, contoh penggunaan tag title:

```
<html>
<head>
<title> Contoh Pembuatan Judul Homepage </title>
</head>
</html>
```

- **Tag META**

Tag META, digunakan untuk menunjukkan informasi metadata mengenai halaman web yang sedang diakses, contoh penggunaan tag meta:

```
<html>
<head>
<meta name="generator" content="Wordpress 3.0"/>
</head>
</html>
```

- **Tag STYLE**

Tag STYLE, digunakan untuk menentukan berbagai bentuk, ukuran, warna, dari tag yang digunakan agar tidak terjadi pemborosan penggunaan tag. Berikut contoh tag STYLE:

```
<html>
<head>
<style type="text/css">
Body {background-color:orange}
</style>
</head>
</html>
```

- **Tag LINK**

Tag LINK, digunakan untuk menautkan suatu halaman dengan halaman atau dokumen lainnya. Contoh penggunaan tag LINK:

```
<html>
<head>
<a href="http://www.tastajaya.blogspot.com"> Blog
Saya </a>
</head>
</html>
```

- **Tag BASE**

Tag BASE, menyatakan URL asal dari suatu dokumen

Contoh:

```
<HTML>
<HEAD>
<TITLE>Homepage Saya</TITLE>
<BASE HREF=http://www.tastajaya.blogspot.com>
</HEAD>
</HTML>
```

## b) **BODY**

Tag Body, merupakan tempat untuk menuliskan isi dari sebuah halaman web seperti teks, gambar, tabel dan lainnya yang akan ditampilkan pada halaman web. Tag Body memiliki beberapa atribut diantaranya:

- Tag `<background=" " />`, digunakan untuk memuat latar belakang dengan gambar/foto.

Contoh:

```
<HTML>
<HEAD>
<TITLE>Contoh Penggunaan Tag Background</TITLE>
</HEAD>
<body background="E:\Mater Ajar\Desert.jpg" >
</body>
</HTML>
```

- Tag `<bgcolor=" " />`, digunakan untuk menentukan warna latar belakang homepage.

Contoh:

```
<HTML>
<HEAD>
<TITLE>Contoh Penggunaan Tag BgColor</TITLE>
</HEAD>
<body bgcolor="#FFFFBB" >
</body>
</HTML>
```

- Tag `<bgproperties=" " />`, digunakan untuk menentukan nilai property background tetap

Contoh:

```
<HTML>
<HEAD>
<TITLE>Contoh Penggunaan Tag BgProperties</TITLE>
</HEAD>
<body bgproperties="fixed" >
</body>
</HTML>
```

- Tag `<leftmargin=" " />`, digunakan untuk menentukan batas kiri halaman dalam satuan pixel.

Contoh:

```
<HTML>
<HEAD>
<TITLE>Contoh Penggunaan Tag Leftmargin</TITLE>
</HEAD>
<body leftmargin="50" bgcolor="#99CC33">
Menentukan Batas Kiri Suatu Halaman Web dalam
satuan pixel
</body>
</HTML>
```

- Tag `<topmargin=" " />`, digunakan untuk menentukan batas atas halaman dalam satuan pixel

Contoh:

```
<HTML>
<HEAD>
<TITLE>Contoh Penggunaan Tag TopMargin</TITLE>
</HEAD>
<body leftmargin="100" topmargin="150"
bgcolor="#99CCAA">
Menentukan batas atas halaman dalam satuan pixel
</body>
</HTML>
```

- Tag `<text=" " />`, digunakan untuk menentukan warna teks,

Contoh :

```
<HTML>
<HEAD>
<TITLE>Contoh Penggunaan Tag Text</TITLE>
</HEAD>
<body leftmargin="100" topmargin="150"
bgcolor="#99CCAA" text="#FF0000">
Menentukan Warna Teks
</body>
</HTML>
```



## **PEMROGRAMAN PYTHON**

### **Pengenalan Python**

Python merupakan salah satu bahasa pemrograman dan penunjang pada perkembangan FOSS dan Linux. Tidak hanya kedua itu, di perkembangan teknologi lainnya, Python hadir sebagai salah satu teknologi mumpuni yang patut kita pelajari. Biasanya Python sudah tertanam di distro - distro tertentu seperti Ubuntu, Fedora, dan Slackware. Python sendiri merupakan bahasa pemrograman yang mendukung paradigma object oriented programming ataupun scripting. Python hadir di ranah sistem operasi, virtualisasi, jaringan komputer, grafika komputer, kecerdasan buatan, teknologi web, game, dan lain sebagainya. Bahasa pemrograman ini dibuat oleh Guido van Rossum dari Amsterdam, Belanda. Pada awalnya, motivasi pembuatan bahasa pemrograman ini adalah untuk bahasa skrip tingkat tinggi pada sistem operasi terdistribusi Amoeba. Bahasa pemrograman ini menjadi umum digunakan untuk kalangan engineer seluruh dunia dalam pembuatan perangkat lunaknya, bahkan beberapa perusahaan menggunakan python sebagai pembuat perangkat lunak komersial.

## Bahasa Pemrograman Python

Python merupakan bahasa pemrograman tingkat tinggi (High Level Language). Python merupakan bahasa pemrograman yang menduduki peringkat ke-5 sebagai bahasa pemrograman yang paling banyak digunakan di seluruh dunia. Setiap bahasa pemrograman pastilah memiliki kelebihan dan kekurangan, Python juga tidak terlepas dari kelebihan dan kekurangan. Berikut ini beberapa kelebihan bahasa pemrograman Python :

- a) Python merupakan bahasa pemrograman yang mudah dipelajari dan cepat dalam membantu programmer dalam membuat aplikasi, baik aplikasi dalam bentuk prototype maupun aplikasi yang siap digunakan oleh user.
- b) Python menganut konsep OOP (Object Oriented Programming).
- c) Coding yang dibuat dalam bahasa Python gampang dibaca oleh programmer. Karena membaca coding Python seperti membaca kalimat dalam Bahasa Inggris.

Kekurangan bahasa pemrograman Python adalah kecepatan menjalankan aplikasi yang tidak secepat aplikasi yang dibuat dengan menggunakan bahasa pemrograman C atau C++.

## Variabel dan Tipe Data dan Operator

Operator, Variabel, dan Tipe Data merupakan konsep dasar pemrograman Python yang membantu memahami dan mengembangkan program. Nilai-nilai dapat disimpan dalam variabel dan diberi nama yang terkait dengannya. Tipe data menentukan jenis nilai yang dapat disimpan dalam variabel, seperti teks, bilangan bulat, bilangan desimal, atau boolean. Operator dapat melakukan operasi matematika, perbandingan, atau logika pada data yang disimpan. Memahami ide ini memungkinkan kita untuk memanipulasi dan memproses data dalam program Python dengan lebih efisien, pada subbab ini kita

akan membahas tentang penggunaan variable, tipe data, dan operator.

## **Variabel**

Variabel adalah tempat di mana data atau nilai tertentu dapat disimpan dan digunakan. Variabel digunakan untuk memberi nama pada data yang disimpan agar dapat digunakan dan dimodifikasi dalam program.

## **Deklarasi Variabel**

Variabel dapat didefinisikan dengan memberikan variabel nama yang unik dan menginisialisasi dengan nilai tertentu. Nama variabel harus mengikuti aturan tertentu, mis. Misalnya. mereka tidak boleh dimulai dengan angka, tidak boleh mengandung spasi atau karakter khusus tertentu, dan bersifat case-sensitive.

Perhatikan contoh kode berikut :

```
jarakTempuh    = 10 #penamaan Variabel yang benar
jarak Tempuh   = 10 #penamaan Variabel yang salah
1jarakTempuh  = 10 #penamaan Variabel yang salah
```

Pada kode diatas adalah bebrapa dklarasi variable dengan penamaan yang berbeda beda, dimana pada variabel yang pertama adalah contoh variabel yang benar, dan pada variabel yang kedua adalah penamaan variabel yang salah karena menggunakan spasi untuk memisahkan kata pada variabel yang seharusnya dilarang menggunakan spasi, lalu variabel yang ketiga juga penamaan variabel yang salah karena menggunakan angka pada didepan nama variabel yang seharusnya dilarang menggunakan angka di depan pada penamaan variable.

## **Inisialisasi Variabel**

Dalam Python, variabel dapat diinisialisasi dengan memberikan nama variabel yang unik dan mengisinya dengan nilai tertentu. Proses inisialisasi variabel dilakukan dengan



menggunakan tanda sama dengan (=) untuk memberikan nilai pada variabel.

Berikut adalah beberapa contoh inisialisasi variabel dalam Python:

```
# Inisialisasi variabel dengan bilangan bulat
umur = 25
jumlah_murid = 30

# Inisialisasi variabel dengan bilangan desimal
nilai_pi = 3.14
harga_barang = 9.99

# Inisialisasi variabel dengan teks
nama = "John Doe"
alamat = 'Jl. Raya No. 123'

# Inisialisasi variabel dengan tipe data boolean
is_active = True
is_available = False

# Inisialisasi variabel dengan tipe data kompleks (misalnya, daftar atau kamus)
daftar_angka = [1, 2, 3, 4, 5]
data_mahasiswa = {'nama': 'John', 'umur': 20, 'jurusan': 'Informatika'}
```

Dalam contoh-contoh di atas, variabel diinisialisasi dengan nama yang sesuai dan diberikan nilai yang sesuai dengan tipe data yang diinginkan. Variabel dapat diinisialisasi secara langsung

dengan nilai yang diberikan, seperti bilangan bulat, bilangan desimal, teks, atau dengan nilai dari tipe data lain, seperti daftar atau kamus.

Variabel di Python tidak perlu dideklarasikan sebelum penggunaan. Mereka akan secara otomatis dibuat ketika nilai diberikan ke dalamnya. Selain itu, nilai dalam variabel dapat diubah atau dimodifikasi sepanjang program dengan memberikan nilai baru.

## **Tipe Data**

Pada subbab sebelumnya sebenarnya kita sudah meyinggung dan menggunakan tipe data tetapi kita belum mengenal sepenuhnya mengenai tipe data walau tidak nampak, pada subbab ini kita akan mengenal lebih jauh mengenai tipe data

Python menyediakan beragam tipe data yang mendukung berbagai jenis informasi, mulai dari angka, teks, hingga struktur data yang lebih kompleks. Setiap tipe data memiliki karakteristik dan perilaku tertentu yang memungkinkan kita untuk memanipulasi data dengan cara yang sesuai.

Mari kita mulai dengan menjelajahi berbagai tipe data dalam Python dan cara menggunakan mereka untuk memproses informasi dengan tepat. Dengan pemahaman yang solid tentang tipe data, Anda akan siap untuk membangun program yang kuat dan efektif dalam bahasa pemrograman Python.

### **1. Tipe Data Angka**

Dalam bahasa pemrograman Python, terdapat beberapa tipe data yang digunakan untuk menyimpan nilai angka. Tipe data angka ini mencakup bilangan bulat (integer), bilangan desimal (float), dan bilangan kompleks (complex). Mari kita bahas lebih lanjut tentang masing-masing tipe data angka beserta contoh penggunaannya.

#### **a. Tipe data Integer**

Tipe data integer (int) digunakan untuk menyimpan bilangan bulat.

Contohnya:

umur = 21

jumlah\_siswa = 30

Dalam contoh di atas, variabel umur dan jumlah\_siswa dideklarasikan dengan tipe data integer dan diinisialisasi dengan bilangan bulat.

### **b. Tipe data Float**

Tipe data float (float) digunakan untuk menyimpan bilangan desimal atau pecahan. Contohnya:

nilai\_pi = 3.14

tinggi = 1.75

Dalam contoh di atas, variabel nilai\_pi dan tinggi dideklarasikan dengan tipe data float dan diinisialisasi dengan bilangan desimal.

### **c. Tipe data Complex**

Tipe data complex (complex) digunakan untuk menyimpan bilangan kompleks dengan bagian real dan imajiner. Contohnya:

bilangan\_complex = 2 + 3j

Dalam contoh di atas, variabel bilangan\_complex dideklarasikan dengan tipe data complex dan diinisialisasi dengan bilangan kompleks.

## **2. Tipe Data Teks**

Dalam pemrograman Python, tipe data teks (str) digunakan untuk menyimpan teks atau urutan karakter. Tipe data ini sangat berguna dalam memanipulasi dan mengolah data teks, seperti string, kata-kata, kalimat, atau paragraph.

### **a. Tipe data String**

Tipe data string digunakan untuk menyimpan teks. Teks diapit oleh tanda kutip tunggal (') atau ganda ("").

Contohnya:

```
nama = "John Doe"  
alamat = 'Jl. Raya No. 123'
```

Dalam contoh di atas, variabel `nama` dan `alamat` dideklarasikan dengan tipe data `string` dan diinisialisasi dengan teks.

### 3. Tipe Boolean

Tipe data boolean (`bool`) adalah tipe data yang menyimpan nilai kebenaran. Nilai `True` menunjukkan pernyataan yang benar, sedangkan nilai `False` menunjukkan pernyataan yang salah. Contoh penggunaan tipe data boolean:

```
is_active = True  
is_available = False
```

Dalam contoh di atas, variabel `is_active` diberi nilai `True`, sedangkan `is_available` diberi nilai `False`.

### 4. Tipe Data Sekuensial

Dalam pemrograman Python, tipe data sekuensial digunakan untuk menyimpan dan mengelola kumpulan nilai atau elemen dalam urutan tertentu. Tipe data sekuensial ini mencakup `list`, `tuple`, dan `range`. Mari kita bahas lebih lanjut tentang masing-masing tipe data sekuensial beserta contoh penggunaannya.

#### a. List

List adalah tipe data sekuensial yang digunakan untuk menyimpan kumpulan nilai yang dapat diubah (`mutable`) dan diakses berdasarkan indeks. Contohnya:

```
angka = [1, 2, 3, 4, 5]  
hewan = ['anjing', 'kucing', 'tikus']
```

Dalam contoh di atas, variabel `angka` dan `hewan` dideklarasikan sebagai `list` dan diinisialisasi dengan kumpulan nilai.

Beberapa operasi yang dapat dilakukan pada tipe data list antara lain:

- Mengakses elemen: `angka[0]` akan mengembalikan nilai pertama dalam list angka.
- Memotong list: `angka[1:3]` akan mengembalikan sublist dari indeks 1 hingga 2.
- Menambah elemen: `hewan.append('gajah')` akan menambahkan elemen 'gajah' ke list hewan.
- Mengubah elemen: `angka[2] = 6` akan mengubah nilai elemen ke-3 dalam list angka menjadi 6.
- Menghapus elemen: `del buah[0]` akan menghapus elemen pertama dalam list hewan.

## b. Tuple

Tuple adalah tipe data sekuensial yang digunakan untuk menyimpan kumpulan nilai yang tidak dapat diubah (immutable) setelah diinisialisasi.

Contohnya:

```
koordinat = (3, 4)
```

Dalam contoh di atas, variabel `koordinat` dan `warna` dideklarasikan sebagai tuple dan diinisialisasi dengan kumpulan nilai.

Tuple memiliki karakteristik yang mirip dengan list, tetapi tidak dapat diubah setelah diinisialisasi. Oleh karena itu, tidak ada operasi untuk menambah atau mengubah elemen tuple. Namun, kita dapat mengakses elemen tuple dengan menggunakan indeks, seperti `koordinat[0]` untuk mendapatkan nilai pertama dalam tuple `koordinat`.

## c. Range

Range adalah tipe data sekuensial yang digunakan untuk menghasilkan rentang bilangan dalam suatu interval.

Contohnya:

```
bilangan = range(1, 6)
```

Dalam contoh di atas, variabel bilangan diinisialisasi dengan range bilangan 1 hingga 5.

Range digunakan terutama dalam pengulangan (loop) untuk menghasilkan urutan bilangan. Misalnya, kita dapat menggunakan for loop untuk mengiterasi melalui range :

```
for i in range(1, 6):  
    print(i)
```

Dalam contoh di atas, range(1, 6) akan menghasilkan urutan bilangan 1, 2, 3, 4, dan 5, dan loop akan mencetak nilai-nilai tersebut.

## Sintaks dasar Python

### Membuat dan Menjalankan Python

Kode python perlu disimpan kedalam file dengan ekstensi .py untuk menuliskan kode python anda dapat menggunakan editor apa saja seperti visual studio code yang sudah di pasang pada bab sebelumnya. Berikut adalah Langkah Langkah pembuatan aplikasi console dengan menggunakan python yang akan menampilkan teks "Hello World" ke terminal :

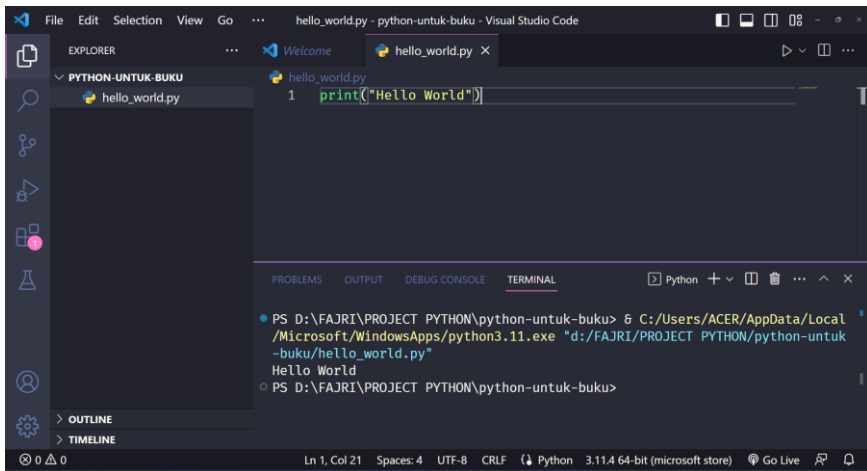
1. Buatlah Direktori kerja, misalnya D:/projectpython
2. Jalankan Visual Studio Code lalu tulis kode berikut ini:  

```
print("Hello World")
```
3. Simpan file dengan ekstensi .py (contoh : hello\_world.py) dan tempatkan file ke dalam direktori kerja yang sudah dibuat sebelumnya

Untuk menjalankan aplikasi, gunakan perintah python namafile di dalm direkori kerja, seperti berikut:

```
cd D:/projectpython  
python hello_world.py
```

Jika anda menggunakan teks editor visual studio code untuk menuliskan kode diatas, anda dapat jula langsung menjalankannya dengan menekan tombol shortcut F5 pada file yang ingin di eksekusi, seperti yang ditunjukkan oleh gambar di bawah ini :



**Gambar 31. Hello World**

## Sintaks Pada Python

Perintah pemrograman adalah instruksi atau langkah khusus yang diberikan ke komputer dalam bahasa pemrograman untuk melakukan tugas tertentu. Setiap bahasa pemrograman memiliki sintaks dan aturan khusus untuk menulis perintah program.

Perintah program dapat mencakup operasi matematika, manipulasi data, pengambilan keputusan, loop, pemanggilan fungsi, dan banyak lagi. Tujuan instruksi program adalah untuk menginstruksikan komputer untuk melakukan tugas yang diinginkan atau memecahkan masalah tertentu.

Contoh perintah program dalam Python:

```
x = 20
y = 8
z = x - y
print("Hasil pengurangan x dan y
adalah:", z)
```

Pada contoh di atas, perintah-perintah program meliputi penugasan nilai ke variabel `x` dan `y`, melakukan operasi

penjumlahan antara x dan y, dan menampilkan hasilnya ke layar menggunakan perintah **print**.

### **Blok Pada Pemrograman Phyton**

Blok program adalah sekumpulan instruksi program yang dikelompokkan bersama dalam satu blok tunggal. Dalam berbagai bahasa pemrograman, contohnya Python, blok program digunakan untuk menggabungkan beberapa instruksi program agar dieksekusi secara bersamaan.

Umumnya, blok program didefinisikan oleh suatu struktur kendali khusus seperti loop atau pernyataan pengambilan keputusan (if-else). Blok program terdiri dari satu atau lebih baris kode yang dikenali berdasarkan indentasi yang sama atau lebar yang serupa.

Contoh penggunaan blok program dalam Python:

```
if x > 3:
    print("Nilai x lebih besar dari 3.")
    print("contoh blok program didalam
if statement")
```

Pada contoh di atas, kita memiliki blok program yang dimulai dengan pernyataan if dan diakhiri dengan pernyataan print yang berada di luar blok. Semua perintah program yang berada di dalam blok if akan dieksekusi hanya jika kondisi  $x > 3$  bernilai benar (True).

### **Komentar Pada Pemrograman Phyton**

Komentar program di Python adalah potongan kode yang tidak dieksekusi oleh juru bahasa Python. Mereka digunakan untuk memberikan penjelasan kode, dokumentasi, atau catatan singkat untuk pengembang atau pembaca kode. Komentar program tidak mempengaruhi pengoperasian program dan tidak akan mempengaruhi produk atau hasil program.

Komentar Python biasanya digunakan:



- Memberikan penjelasan tentang fungsi, tujuan atau algoritma kode.
- Buat catatan atau pengingat untuk pengembang tentang kode tertentu.
- Tinggalkan instruksi atau catatan untuk pengembang lain yang mungkin bekerja dengan kode tersebut.

Komentar Python dimulai dengan tanda hash (#) dan dapat ditempatkan di awal baris atau setelah kode sebenarnya. Penerjemah Python mengabaikan teks apa pun setelah tanda hashtag(#).

Contoh penggunaan komentar dalam Python:

```
# Ini adalah komentar dalam Python

# Catatan - Penjelasan variabel
nama = "John" #Instruksi - Variabel untuk
menyimpan nama pengguna

# Catatan - Perhitungan
hasil = 10 + 5 # Instruksi - Penjumlahan
dua angka

# Pemanggilan fungsi
# fungsi_tertentu(argumen)
```

Pada contoh di atas, komentar digunakan untuk memberikan penjelasan umum, memberikan catatan tentang variabel, memberikan instruksi tentang perhitungan, dan memberikan contoh pemanggilan fungsi.

## Operasi Matematika

Operator merupakan simbol khusus yang disediakan oleh python untuk melakukan operasi operasi tertentu seperti perhitungan bilangan, penyambungan teks, perbandingan dua buah bilangan dan sebagainya sub-bab ini akan membahas tentang operator-operator yang disediakan oleh python.

### Operator Penugasan

Operator penugasan adalah operator yang digunakan untuk memberikan nilai pada suatu variabel. Operator ini memungkinkan kita untuk mengubah atau memodifikasi nilai variabel dengan menggunakan operasi matematika atau operasi lainnya. Berikut ini adalah contoh operator penugasan dalam Python:

```
x = 10
y = x + 5
```

Pada kode program diatas menunjukkan operator penugasan, dimana variabel x diisi dengan nilai 10 .lebih detailnya lagi, perintah tersebut menyatakan bahwa opertaor = nenugaskan variabel /referensi X untuk menunjuk ke alamat memoryyang didalamnya berisi objek bertipe int dengan nilai 10, dan dimana variabel x dapat dipanggil/digunakan oleh variabel yang lainnya.

### Operator Artimatika

Operator aritmatika digunakan dalam pemrograman Python untuk melakukan operasi matematika pada bilangan atau ekspresi numerik. Dalam Python, terdapat beberapa operator aritmatika yang dapat digunakan untuk melakukan penjumlahan, pengurangan, perkalian, pembagian, modulus, dan lainnya.

Operator	Keterangan
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
//	Pembagian Bulat
%	Modulus (Sisa Bagi)
**	Pangkat

Dalam tabel di atas, kolom "Operator" menunjukkan operator aritmatika yang digunakan, kolom "Keterangan" menjelaskan fungsi dari operator tersebut.

Operator aritmatika ini memungkinkan kita untuk melakukan berbagai operasi matematika pada bilangan atau ekspresi numerik dalam bahasa pemrograman Python. Dengan memahami penggunaan dan keterangan masing-masing operator, kita dapat melakukan perhitungan matematika yang diperlukan dalam program dengan tepat dan akurat.

Berikut adalah contoh penggunaan program yang menunjukkan penggunaan operator operator aritmatika:

```
# operasi aritmatika

a = 10
b = 5

# operasi penjumlahan
hasil = a + b
print(a, " + ", b, " = ", hasil)

# operasi pengurangan
hasil = a - b
print(a, " - ", b, " = ", hasil)
```

```
# operasi perkalian
hasil = a * b
print(a, " * ", b, " = ", hasil)

# operasi pembagian
# pada python hasil bagi otomatis menjadi tipe
data float
hasil = a / b
print(a, " / ", b, " = ", hasil)

# operasi modulus - sisa pembagian
hasil = a % b
print(a, " % ", b, " = ", hasil)

# operasi floor division - dibulatkan kebawah
hasil = a // b
print(a, " // ", b, " = ", hasil)

# operasi eksponen(pangkat)
hasil = a ** b
print(a, " ** ", b, " = ", hasil)
```

Output Program:

```
-----
10 + 5 = 15
10 - 5 = 5
10 * 5 = 50
10 / 5 = 2.0
10 % 5 = 0
10 // 5 = 2
10 ** 5 = 100000
-----
```

## Penggabungan Operator Penugasan dengan Operator Aritmatika

Dalam bahasa pemrograman Python, kita dapat menggabungkan operator penugasan dengan operator aritmatika untuk melakukan operasi matematika dan memperbarui nilai variabel secara bersamaan. Dengan menggabungkan keduanya, kita dapat menghemat penulisan kode yang berulang dan membuat kode lebih efisien. Berikut adalah contoh penggabungan operator penugasan dengan operator aritmatika yang umum digunakan:

- `%=`: Operator ini digunakan untuk menghitung sisa bagi dan mengupdate nilainya.  
Contoh:  
`x = 10`  
`x %= 3 # sama dengan x = x % 3`
- `//=`: Operator ini digunakan untuk menghitung hasil bagi bulat dan mengupdate nilainya.  
Contoh:  
`x = 10`  
`x //= 3 # sama dengan x = x // 3`
- `**=`: Operator ini digunakan untuk menghitung pangkat dan mengupdate nilainya.  
Contoh:  
`x = 2`  
`x **= 3 # sama dengan x = x ** 3`

Dengan menggabungkan operator penugasan dengan operator aritmatika, kita dapat melakukan operasi matematika dan mengupdate nilai variabel dalam satu pernyataan. Hal ini memungkinkan kita untuk menulis kode yang lebih ringkas dan efisien. Penggunaan penggabungan ini sangat berguna dalam

banyak skenario pemrograman, terutama saat melakukan iterasi atau pembaruan nilai variabel secara berulang.

### Operator Komparasi

Operator komparasi digunakan dalam pemrograman Python untuk membandingkan dua nilai atau ekspresi. Hasil dari operasi perbandingan adalah nilai boolean, yaitu True (benar) atau False (salah), yang menunjukkan hasil dari perbandingan tersebut. Operator komparasi memungkinkan kita untuk menguji kesetaraan, ketidaksetaraan, perbandingan nilai, dan hubungan antara dua nilai.

Berikut adalah tabel operator komparasi yang umum digunakan dalam Python:

Operator	Keterangan
==	Sama dengan
!=	Tidak sama dengan
>	Lebih besar dari
<	Lebih kecil dari
>=	Lebih besar atau sama dengan
<=	Lebih kecil atau sama dengan
is	Membandingkan memory

Operator komparasi banyak digunakan untuk membentuk kondisi pada konstruksi pemilihan maupun pengulangan. Contoh program di bawah ini akan menunjukkan operator operator diatas :

```
a = 3
b = 2

# lebih besar dari ( > )
print("==== Lebih Besar dari ( > ) =====")
hasil = a > 2
print(a, " > ", 2, " = ", hasil)
hasil = b > 2
print(b, " > ", 2, " = ", hasil)
```

```

hasil = a > 4
print(a, " > ", 4, " = ", hasil)

# lebih kecil dari ( < )
print("==== Lebih kecil dari ( < ) =====")
hasil = a < 2
print(a, " < ", 2, " = ", hasil)
hasil = b < 2
print(b, " < ", 2, " = ", hasil)
hasil = a < 4
print(a, " < ", 4, " = ", hasil)

# lebih besar sama dengan ( >= )
print("==== lebih besar sama dengan ( >= ) =====")
hasil = a >= 2
print(a, " >= ", 2, " = ", hasil)
hasil = b >= 2
print(b, " >= ", 2, " = ", hasil)
hasil = a >= 4
print(a, " >= ", 4, " = ", hasil)

# lebih Kecil sama dengan ( <= )
print("==== lebih Kecil sama dengan ( <= ) =====")
hasil = a <= 2
print(a, " <= ", 2, " = ", hasil)
hasil = b <= 2
print(b, " <= ", 2, " = ", hasil)
hasil = a <= 4
print(a, " <= ", 4, " = ", hasil)

# sama dengan sama dengan ( == )
print("==== sama dengan sama dengan ( == ) =====")
hasil = a == 2
print(a, " == ", 2, " = ", hasil)

```

```

hasil = b == 2
print(b, " == ", 2, " = ", hasil)
hasil = a == 4
print(a, " == ", 4, " = ", hasil)

# tidak sama dengan ( <= )
print("===== tidak sama dengan ( != ) =====")
hasil = a != 2
print(a, " != ", 2, " = ", hasil)
hasil = b != 2
print(b, " != ", 2, " = ", hasil)
hasil = a != 4
print(a, " != ", 4, " = ", hasil)

# is -> membandingkan objek/memory
print("===== is =====")
hasil = a is b
print(a, " is ", b, " = ", hasil)

```

Output Program:

```

-----
===== Lebih Besar dari ( > ) =====
3 > 2 = True
2 > 2 = False
3 > 4 = False
===== Lebih kecil dari ( < ) =====
3 < 2 = False
2 < 2 = False
3 < 4 = True
===== lebih besar sama dengan ( >= ) =====
3 >= 2 = True
2 >= 2 = True
3 >= 4 = False
===== lebih Kecil sama dengan ( <= ) =====

```



```

3 <= 2 = False
2 <= 2 = True
3 <= 4 = True
===== sama dengan sama dengan ( == ) =====
3 == 2 = False
2 == 2 = True
3 == 4 = False
===== tidak sama dengan ( != ) =====
3 != 2 = True
2 != 2 = False
3 != 4 = True
===== is =====
3 is 2 = False
-----

```

Operator komparasi digunakan untuk menguji kondisi dan pengambilan keputusan dalam program. Hasil dari operasi perbandingan sering digunakan dalam pernyataan pengendali seperti if, while, dan lainnya untuk mengontrol alur program. Dengan memahami dan menggunakan operator komparasi dengan tepat, kita dapat membuat program yang lebih fleksibel dan responsif terhadap kondisi yang diberikan.

Berikut Contoh program operator komparasi dengan menggunakan operator pemilihan:

```

nilai = 85
kkm = 80

if nilai > kkm :
    print(f"nilai kamu {nilai} diatas {kkm}")
    print("Ket : Kamu Lulus")
else :
    print(f"nilai kamu ${nilai} dibawah ${kkm}")
    print ("Kamu tidak Lulus")

```

Output Program:

---

nilai kamu 85 diatas 80  
Ket : Kamu Lulus

---

Pada contoh kode diatas kita menggunakan logika percabangan dengan memanfaatkan operator komparasi untuk menentukan hasil dari nilai tersebut diamana akan mendapatkan hasil yaitu keterangan kelulusan

Berikut juga Contoh program operator komparasi dengan menggunakan logika perulangan:

```
n = 1
while n <= 5 :
    print(f"ini adalah perulangan ke-{n}")
    n+=1
```

Output Program:

---

ini adalah perulangan ke-1  
ini adalah perulangan ke-2  
ini adalah perulangan ke-3  
ini adalah perulangan ke-4  
ini adalah perulangan ke-5

---

Dari contoh diatas kita menggunakan logika perulangan dengan memanfaatkan operator komparasi untuk mendapatkan output program yang berulang sampai kondisi terpenuhi.

### **Operator Logika**

Operator logika digunakan dalam pemrograman Python untuk menggabungkan atau memanipulasi nilai kebenaran (boolean). Operator ini memungkinkan kita untuk menggabungkan kondisi dan menghasilkan nilai boolean berdasarkan logika

Boolean. Operator logika yang umum digunakan dalam Python meliputi and (dan), or (atau), dan not (bukan). Berikut ini adalah penjelasan tentang masing-masing operator logika beserta contoh penggunaannya:

- Operator Logika AND (and): Operator and menghasilkan nilai True jika kedua operand atau kondisi yang diberikan bernilai True. Jika salah satu atau kedua operand bernilai False, maka hasilnya adalah False.

Bisa kita lihat pada tabel kebenaran dibawah ini

A	B	A and B
True	True	True
True	False	False
False	True	False
False	False	False

- Operator Logika OR (or): Operator or menghasilkan nilai True jika salah satu atau kedua operand atau kondisi yang diberikan bernilai True. Hanya jika kedua operand atau kondisi tersebut bernilai False, maka hasilnya adalah False.

Bisa kita lihat pada tabel kebenaran dibawah ini

A	B	A or B
True	True	True
True	False	True
False	True	True
False	False	False

- Operator Logika NOT (not): Operator not digunakan untuk membalikkan nilai kebenaran (boolean) dari suatu operand atau kondisi. Jika operand awal bernilai True, maka hasilnya adalah False. Jika operand awal bernilai False, maka hasilnya adalah True.

Bisa kita lihat pada tabel kebenaran dibawah ini

A	not A
True	False
False	True

Berikut ini adalah contoh kode untuk menjelaskan cara kerja operator logika:

```
A = True
```

```
B = False
```

```
# Logika OR
```

```
print("==== OR =====")
```

```
print(A, " or ", B, " = ", A or B)
```

```
# Logika AND
```

```
print("==== AND =====")
```

```
print(A, " and ", B, " = ", A and B)
```

```
# Logika XOR
```

```
print("==== XOR =====")
```

```
print(A, " xor ", B, " = ", A ^ B)
```

```
# Logika NOT
```

```
print("==== NOT =====")
```

```
print("NOT ", A, " = ", not A)
```

```
print("NOT ", B, " = ", not B)
```

Output Program:

```
-----
```

```
==== OR =====
```

```
True or False = True
```

```
==== AND =====
```

```
True and False = False
```

```
==== XOR =====
```

```
True xor False = True
```

```
===== NOT =====  
NOT True = False  
NOT False = True
```

---

Operator logika digunakan untuk menggabungkan atau memanipulasi nilai kebenaran dan sering digunakan dalam struktur kendali (misalnya, if, while) untuk mengendalikan alur program berdasarkan kondisi yang diberikan. Dengan menggunakan operator logika, kita dapat membuat program yang lebih kompleks dan dapat menangani banyak kondisi yang berbeda.

### Operator Pada String

Operator pada string digunakan untuk melakukan operasi seperti penggabungan string, pengulangan string, akses karakter individu, pemotongan string, dan lainnya. Berikut adalah beberapa operator string yang umum digunakan dalam Python:

- **Operator Penambahan (+):** Operator penambahan digunakan untuk menggabungkan atau menyatukan dua string menjadi satu string baru.

Contoh:

```
nama_depan = "Fajri"  
nama_belakang = "Chan"  
nama_lengkap = nama_depan + " " +  
nama_belakang  
print(nama_lengkap)
```

Output Program:

---

Fajri Chan

---

- **Operator Perulangan ():** *Operator perulangan* digunakan untuk mengulang string sejumlah tertentu.

Contoh:

```
kata = "Halo"  
ulang_kata = kata * 3  
print(ulang_kata)
```

Output Program:

---

```
HaloHaloHalo
```

---

- Operator Indeks `[]`: Operator indeks digunakan untuk mengakses karakter individual dalam string berdasarkan posisi indeksnya.

Contoh:

```
kata = "Hello"  
print(f"huruf dari indeks ke 0 dari {kata}  
adalah", kata[0])  
print(f"huruf dari indeks ke 3 dari {kata}  
adalah", kata[3])
```

Output Program:

---

```
huruf dari indeks ke 0 dari Hello adalah H  
huruf dari indeks ke 3 dari Hello adalah l
```

---

- Operator Slicing `[:]`: Operator slicing digunakan untuk memotong (mengambil sebagian) string berdasarkan rentang indeks tertentu.

Contoh:

```
kata = "Buku Python"  
potongan = kata[5:11]  
print(potongan)
```

Output Program:

---

```
python
```

---

- Operator Panjang (len()): Operator len() digunakan untuk mengembalikan panjang string (jumlah karakter dalam string).

Contoh:

```
kata = "python"  
panjang = len(kata)  
print(panjang)
```

Output Program:

```
-----  
6  
-----
```

- Operator Keanggotaan (in): Operator keanggotaan (in) digunakan untuk memeriksa apakah karakter atau substring tertentu ada dalam string.

Contoh:

```
kata = "python"  
hasil = "on" in kata  
print(hasil)
```

Output Program:

```
-----  
true  
-----
```

## Struktur Kontrol : Percabangan dan Perulangan

Struktur kontrol adalah suatu konstruksi atau blok program yang berguna untuk mengendalikan jalannya alur program. Secara umum, struktur kontrol itu sendiri dibedakan menjadi tiga jenis, yaitu: struktur pemilihan, struktur pengulangan, dan struktur untuk penanganan eksepsi. Dalam bab ini kita akan membahas struktur kontrol tersebut.

## Struktur Pemilihan

Struktur pemilihan merupakan blok program yang berfungsi untuk memilih perintah yang akan dilakukan oleh program berdasarkan kondisi tertentu yang didefinisikan di dalamnya. Struktur kontrol pemilihan ini memungkinkan kita untuk menjalankan blok kode tertentu hanya jika kondisi yang diberikan terpenuhi.

Pembuatan struktur pemilihan dalam bahasa python hanya bisa menggunakan perintah if ada beberapa perintah if dalam python, yaitu :

- Perintah if satu kasus
- Perintah if dua kasus
- Perintah if tiga kasus atau lebih
- Perintah if didalam if (nested if)

### Perintah If Satu Kasus

Pernyataan if adalah salah satu struktur kontrol yang paling dasar dalam Python. Ini digunakan untuk menjalankan blok kode tertentu jika kondisi yang diberikan bernilai True. Dalam satu kasus, hanya ada satu kondisi yang dievaluasi dan hanya ada satu blok kode yang akan dijalankan jika kondisi terpenuhi.

Berikut adalah format umum dari pernyataan if dalam satu kasus :

*if kondisi:*

*pernyataan\_1*

*pernyataan\_2*

*...*

*#Blok Kode akan dijalankan jika kondisi benar (true)*

Perintah pada code diatas hanya dilakukan ketika kondisi benar (true). Jika kondisi bernilai salah (false) aksi akan diabaikan atau tidak dilakukan.

Contoh penggunaan if di satu kasus dapat dilihat pada kode dibawah ini :



```
x = 5
```

```
if x > 0:  
    print("Nilai x adalah positif")  
    print("Ini adalah blok kode dalam satu kasus")
```

Output Program:

---

```
Nilai x adalah positif  
Ini adalah blok kode dalam satu kasus
```

---

Pada contoh di atas, jika nilai  $x$  lebih besar dari 0, maka kedua pernyataan print dalam blok kode akan dijalankan. Jika nilai  $x$  adalah negatif atau nol, blok kode tersebut akan dilewati. Contoh lain dari perintah if satu kasus:

```
nilai = input("Masukkan Nilai :")  
bilangan = int(nilai)  
  
if bilangan % 2 == 0 and bilangan > 0 :  
    print(f"{nilai} adalah bilangan genap")  
    print(f"{nilai} adalah bilangan positif")
```

Output Program:

---

```
Masukkan Nilai :4  
4 adalah bilangan genap  
4 adalah bilangan positif
```

---

Pada contoh kode diatas, kita membuat perintah untuk mengecek apakah nilai yang dimasukkan adalah bilangan genap dan apakah bilangan positif, jika nilainya benar (true) blok kode yang didalamnya akan bekerja dan menapilkannya pada terminal, jika nilainya adalah selain bilangan genap dan selain bilangan positif perintah akan diabaikan.

## Perintah if dua kasus

Pernyataan if-else dengan dua kasus digunakan ketika kita ingin menjalankan dua blok kode yang berbeda tergantung pada hasil evaluasi kondisi yang diberikan. Jika kondisi yang diberikan bernilai True, blok kode dalam pernyataan if akan dijalankan. Jika kondisi bernilai False, blok kode dalam pernyataan else akan dijalankan.

Berikut adalah format umum dari pernyataan if-else dengan dua kasus:

```
if kondisi:
    # Blok kode yang akan dijalankan jika kondisi bernilai
    benar (true)
    pernyataan_1
    pernyataan_2
    ...
else:
    # Blok kode yang akan dijalankan jika kondisi bernilai
    salah (false)
    pernyataan_a
    pernyataan_b
    ...
```

Pada blok kode di atas, kondisi adalah ekspresi atau pernyataan yang dievaluasi. Jika kondisi bernilai True, maka blok kode dalam pernyataan if akan dijalankan. Jika kondisi bernilai False, maka blok kode dalam pernyataan else akan dijalankan.

Contoh penggunaannya dapat dilihat pada program dibawah ini:

```
nilai = input("Masukkan Nilai :")

bilangan = int(nilai)

if bilangan % 2 == 0 :
    print(f"{nilai} adalah bilangan genap")
else :
```

```
print(f"{nilai} adalah bilangan ganjil")
```

Output Program:

---

```
Masukkan Nilai :5
5 adalah bilangan ganjil
```

---

Struktur pemilihan dengan if dua kasus digunakan untuk menangani permasalahan yang memiliki dua kemungkinan. Sebagai contoh code diatas.

Pada kode diatas kita melakukan pengecekan bilangan apakah bilangan itu ganjil atau genap jika kita memasukkan angka bilangan genap if pertama akan bernilai benar (true) dan akan menjalankan pernyataan pada blok programnya, jika angka bilangan yang dimasukkan adalah bilangan ganjil, if pertama akan bernilai salah (false) sehingga blok program di dalam if tidak akan dijalankan dan menuju blok program if yang akan dijalankan.

### Perintah if tiga kasus atau lebih

Pernyataan if-elif-else dengan tiga kasus atau lebih digunakan ketika kita ingin memeriksa beberapa kondisi berbeda dan menjalankan blok kode yang sesuai dengan kondisi yang terpenuhi. elif (singkatan dari "else if") memungkinkan kita menambahkan kondisi tambahan yang akan dievaluasi jika kondisi sebelumnya tidak terpenuhi.

Berikut adalah format umum dari pernyataan if-elif-else dengan tiga kasus atau lebih:

```
if kondisi_1:
    # Blok kode yang akan dijalankan jika kondisi_1 bernilai
    True
    pernyataan_1
    pernyataan_2
    ...
elif kondisi_2:
    # Blok kode yang akan dijalankan jika kondisi_2 bernilai
    True
```

```

    pernyataan_a
    pernyataan_b
    ...
elif kondisi_3:
    # Blok kode yang akan dijalankan jika kondisi_3 bernilai
    True
    pernyataan_x
    pernyataan_y
    ...
else:
    # Blok kode yang akan dijalankan jika semua kondisi
    sebelumnya tidak terpenuhi
    pernyataan_p
    pernyataan_q
    ...

```

Pada blok kode di atas, kondisi\_1, kondisi\_2, dan kondisi\_3 adalah ekspresi atau pernyataan yang dievaluasi secara berurutan. Jika suatu kondisi bernilai True, maka blok kode yang sesuai akan dijalankan. Jika tidak ada kondisi yang bernilai True, maka blok kode dalam pernyataan else akan dijalankan.

```
x = 5
```

```

if x > 0:
    print("Nilai x adalah positif")
    print("Ini adalah blok kode dalam kasus
if")
elif x < 0:
    print("Nilai x adalah negatif")
    print("Ini adalah blok kode dalam kasus
elif")
else:
    print("Nilai x adalah nol")
    print("Ini adalah blok kode dalam kasus
else")

```

Output Program:

---

Nilai x adalah positif  
Ini adalah blok kode dalam kasus if

---

Pada contoh di atas, jika nilai x lebih besar dari 0, maka blok kode dalam pernyataan if akan dijalankan. Jika nilai x lebih kecil dari 0, maka blok kode dalam pernyataan elif akan dijalankan. Jika tidak ada kondisi sebelumnya yang terpenuhi, blok kode dalam pernyataan else akan dijalankan.

Penting untuk memperhatikan indentasi yang benar untuk mengatur blok kode dalam pernyataan if-elif-else. Indentasi yang tepat diperlukan agar blok kode dianggap sebagai bagian dari pernyataan yang sama.

Pernyataan if-elif-else dengan tiga kasus atau lebih memungkinkan kita untuk memeriksa beberapa kondisi dan menjalankan blok kode yang sesuai dengan kondisi yang terpenuhi. Jumlah kondisi dan kasus yang bisa diuji bisa lebih dari tiga, sesuai dengan kebutuhan program.

### **Perintah if didalam if (nested if)**

Pernyataan nested if dalam Python adalah penggunaan struktur if di dalam blok kode if lainnya. Dalam kasus ini, kita dapat mengevaluasi beberapa kondisi secara bertingkat atau berlapis dan menjalankan blok kode yang sesuai dengan setiap kondisi. Berikut adalah format umum dari pernyataan nested if:

```
if kondisi_1:
    # Blok kode yang akan dijalankan jika kondisi_1 bernilai
    True
    if kondisi_2:
        # Blok kode yang akan dijalankan jika kondisi_2 bernilai
        True
        pernyataan_a
        pernyataan_b
```

```

...
else:
    # Blok kode yang akan dijalankan jika kondisi_2 bernilai
False
    pernyataan_c
    pernyataan_d
...
else:
    # Blok kode yang akan dijalankan jika kondisi_1 bernilai
False
    pernyataan_x
    pernyataan_y
...

```

Pada blok kode di atas, kita memiliki struktur if dalam blok kode if lainnya. Jika kondisi kondisi\_1 bernilai True, maka kondisi kondisi\_2 akan dievaluasi. Jika kondisi\_2 juga bernilai True, maka blok kode yang sesuai dengan kondisi\_2 akan dijalankan. Jika kondisi\_2 bernilai False, maka blok kode yang sesuai dengan else dalam nested if akan dijalankan. Jika kondisi\_1 bernilai False, maka blok kode yang sesuai dengan else luar dari nested if akan dijalankan.

Contoh:

```

nilai = input("Masukkan Nilai :")

bilangan = int(nilai)

if bilangan > 0:
    print("Nilai x adalah positif")
    if bilangan % 2 == 0:
        print(f"Nilai {bilangan} adalah
bilangan positif genap")
    else:
        print(f"Nilai {bilangan} adalah
bilangan positif ganjil")

```

```
else:  
    print(f"Nilai {bilangan} adalah negatif  
atau nol")
```

Output Program:

---

```
Masukkan Nilai :10  
Nilai x adalah positif  
Nilai 10 adalah bilangan positif genap
```

---

Pada contoh di atas, kita memiliki nested if. Jika nilai bilangan lebih besar dari 0, maka blok kode dalam pernyataan if pertama akan dijalankan. Kemudian, dalam blok kode tersebut, kita mengevaluasi apakah x adalah bilangan genap atau ganjil. Jika kondisi bilangan  $\% 2 == 0$  bernilai True, maka blok kode dalam pernyataan if kedua akan dijalankan. Jika kondisi bilangan  $\% 2 == 0$  bernilai False, maka blok kode dalam pernyataan else dalam nested if akan dijalankan. Jika nilai bilangan tidak lebih besar dari 0, maka blok kode dalam pernyataan else luar dari nested if akan dijalankan.

Indentasi yang benar sangat penting dalam nested if. Indentasi yang tepat menggambarkan tingkatan blok kode yang sesuai.

## Struktur Perulangan

Struktur pengulangan adalah suatu konstruksi program yang berfungsi untuk melakukan eksekusi perintah secara berulang sampai kondisi tertentu python menyediakan perintah while dan for untuk membentuk struktur pengulangan

## Perintah While

Perintah while dalam Python adalah salah satu struktur kontrol yang digunakan untuk membuat perulangan yang berjalan selama kondisi tertentu bernilai True. Ini memungkinkan kita untuk

menjalankan blok kode berulang kali hingga kondisi berubah menjadi False. Pada setiap iterasi, kondisi akan dievaluasi, dan jika bernilai True, blok kode dalam pernyataan while akan dijalankan. Jika kondisi masih bernilai True, iterasi akan terus berlanjut; jika kondisi menjadi False, perulangan akan berhenti, dan eksekusi program akan melanjutkan ke blok kode setelah pernyataan while.

Bentuk umum penggunaan perintah while adalah sebagai berikut:

```
while kondisi:  
    # Blok kode yang akan diulang selama kondisi bernilai  
True  
    pernyataan_1  
    pernyataan_2  
    ...
```

Selama kondisi bernilai benar di dalam badan pengulangan akan terus dilakukan oleh karena itu harus ada perintah dalam Badan pengulangan yang menyebabkan nilai kondisi berubah menjadi salah agar Proses pengulangan bisa berhenti perhatikan contoh program berikut ini :

```
i = 0  
while i < 5:  
    print(f"ini adalah perulangan yang ke-{i}")  
    i +=1
```

Output Program:

```
-----  
ini adalah perulangan yang ke-0  
ini adalah perulangan yang ke-1  
ini adalah perulangan yang ke-2  
ini adalah perulangan yang ke-3  
ini adalah perulangan yang ke-4  
-----
```

Pada kode diatas variabel i berperan sebagai indeks perulangan yaitu variabel yang digunakan untuk mencatat langkah dalam Proses pengulangan mula-mula variabel diinisialisasikan



dengan nilai 0 selanjutnya program akan memeriksa kondisi yang didefinisikan dalam struktur pengulangan ( $i < 5$ ) karena 0 lebih kecil dari 5 maka kondisi bernilai benar dan program akan mengeksekusi perintah `print(f"ini adalah perulangan yang ke-{i}")`.

Setelah itu nilai dinaikkan 1 melalui perintah `i+=1` sehingga sekarang bernilai 1, nilai 1 ini selanjutnya akan diperiksa lagi di dalam kondisi karena satu juga masih lebih kecil dari 5 maka program akan kembali perkuasi perintah yang terdapat di dalam badan pengulangan dan menaikkan kembali nilai menjadi 2, Demikian seterusnya sampai `i` bernilai 5. ketika variabel `i` bernilai 5 kondisi bernilai salah dan Proses pengulangan akan dihentikan dengan demikian Proses pengulangan diatas dilakukan sebanyak 5 kali mulai dari `i = 0` sampai `i = 4`

### **Perintah *for***

Perintah `for` adalah struktur kontrol yang digunakan dalam Python untuk melakukan pengulangan berdasarkan urutan objek yang dapat diiterasi, seperti list, tuple, string, atau range. Perintah `for` memungkinkan kita untuk menjalankan blok kode berulang kali untuk setiap elemen dalam urutan tersebut.

Berikut adalah format umum dari perintah `for`

*for elemen in urutan:*

*# Blok kode yang akan dijalankan untuk setiap elemen dalam urutan*

*pernyataan\_1*

*pernyataan\_2*

*...*

Pada blok kode di atas, elemen adalah variabel yang akan mewakili setiap elemen dalam urutan saat pengulangan berlangsung. urutan adalah objek yang dapat diiterasi seperti list, tuple, string, atau range.

Contoh program dengan `for`:

```
fruits = ['apple', 'banana', 'mangga']
```

```
for fruit in fruits:  
    print(fruit)
```

Output Program:

```
-----  
apple  
banana  
mangga  
-----
```

Pada contoh di atas, kita memiliki list fruits yang berisi tiga elemen. Dalam perulangan for, variabel fruit mewakili setiap elemen dalam list secara berurutan. Blok kode dalam pernyataan for akan dijalankan untuk setiap elemen, di mana kita mencetak nilai fruit.

Contoh lain program dengan for:

```
for i in range(1, 6):  
    print(i)
```

Output Program:

```
-----  
1  
2  
3  
4  
5  
-----
```

Fungsi range() digunakan untuk menghasilkan urutan bilangan, yang dapat digunakan dalam perulangan for. Kita dapat memberikan parameter mulai, akhir, dan langkah (opsional) untuk mengatur rentang bilangan yang dihasilkan.

## Jump Statement

Pernyataan loncat (jump statements) dalam Python digunakan untuk mengubah aliran eksekusi program dengan

melompati bagian-bagian tertentu dari kode. terdapat tiga jenis pernyataan loncat atau "jump statements" yang digunakan untuk mengubah aliran eksekusi program, yaitu `break`, `continue`, dan `pass`. Meskipun sering disebut sebagai "fungsi", sebenarnya mereka adalah pernyataan (statements) yang mempengaruhi aliran program. Mari kita bahas masing-masing pernyataan loncat tersebut:

- Pernyataan `break`:

Pernyataan `break` digunakan untuk menghentikan perulangan secara paksa. Ketika pernyataan `break` dieksekusi di dalam perulangan, program keluar dari perulangan tersebut dan melanjutkan eksekusi ke blok kode setelah perulangan.

Contoh penggunaan `break` dalam perulangan `for`:

```
for i in range(1, 6):
    if i == 3:
        break
    print(i)
```

Output Program:

```
-----
1
2
-----
```

Pada contoh di atas, ketika `i` bernilai 3, pernyataan `break` dieksekusi, dan perulangan dihentikan secara paksa. Oleh karena itu, hanya nilai 1 dan 2 yang dicetak.

- Pernyataan `continue`:

Pernyataan `continue` digunakan untuk melanjutkan ke iterasi berikutnya dalam perulangan tanpa menjalankan kode yang berada di bawah pernyataan `continue`. Ketika pernyataan `continue` dieksekusi di dalam perulangan, program akan melompati sisa blok kode dalam iterasi saat ini dan melanjutkan ke iterasi berikutnya.

Contoh penggunaan continue dalam perulangan for:

```
for i in range(1, 6):  
    if i == 3:  
        continue  
    print(i)
```

Output Program:

```
-----  
1  
2  
4  
5  
-----
```

Pada contoh di atas, ketika i bernilai 3, pernyataan continue dieksekusi, dan iterasi dengan nilai tersebut dilewati. Oleh karena itu, nilai 3 tidak dicetak.

- **Pernyataan pass**

Pernyataan pass digunakan sebagai tanda tempat yang tidak ada tindakan yang perlu dilakukan dalam suatu blok kode. Ini adalah pernyataan yang kosong dan tidak melakukan apa pun. Pernyataan pass sering digunakan saat kita ingin menyisipkan blok kode yang akan diisi nanti, tetapi sintaksis Python mengharuskan ada kode dalam blok tersebut.

Contoh penggunaan pass dalam pernyataan if:

```
angka = input("Masukkan Nilai : ")  
x = int(angka)
```

```
if x < 0:  
    pass  
else:  
    print("Nilai x positif")
```

Output Program:

```
-----  
Masukkan Nilai : -5
```

---

Pada contoh di atas, jika  $x$  kurang dari 0, pernyataan `pass` dieksekusi, yang berarti tidak ada tindakan khusus yang dilakukan dalam blok tersebut. Namun, jika  $x$  tidak kurang dari 0, maka pernyataan `print` akan dijalankan.

## Fungsi Pada Python

Fungsi pada python adalah kumpulan perintah atau baris kode yang dikelompokkan menjadi satu kesatuan untuk kemudian bisa dipanggil atau digunakan berkali-kali.

Sebuah fungsi bisa menerima parameter, bisa mengembalikan suatu nilai, dan bisa dipanggil berkali-kali secara independen.

Dengan fungsi kita bisa memecah program besar yang kita tulis, menjadi bagian-bagian kecil dengan tugasnya masing-masing.

Juga, fungsi akan membuat kode program kita menjadi lebih “reusable” dan lebih terstruktur.

### Sintaks Fungsi

Dalam python, sintaks pembuatan fungsi terlihat seperti berikut:

```
def <nama_fungsi>(parameters):  
    statements
```

Sintaks di atas secara umum terbagi menjadi 4 bagian:

- Kata kunci `def` yang menjadi pertanda bahwa blok kode program adalah sebuah fungsi
- Nama fungsi yang kita buat
- Parameters yang akan diterima oleh fungsi yang kita buat (tidak wajib)
- Dan blok kode fungsi yang di sana akan kita tulis perintah-perintah yang harus dilakukan oleh sebuah fungsi

Blok kode program di dalam python didefinisikan dengan indentasi. Silakan baca aturan sintaks python untuk lebih lengkapnya.

Contoh sebuah fungsi sederhana dengan nama `halo_dunia()`:

```
def halo_dunia():  
    print('Halo python! Halo dunia!')
```

Fungsi di atas, jika dipanggil, akan mengeksekusi perintah `print()` yang ada di dalamnya.

### Memanggil Fungsi

Berikut ini merupakan cara memanggil fungsi :

Cukup ketik nama fungsinya, ditambah dengan tanda kurung () seperti berikut:

```
halo_dunia()
```

Output:

```
Halo python! Halo dunia
```

Bahkan kita bisa memanggil fungsi `halo_dunia()` berkali-kali:

```
halo_dunia()
```

```
halo_dunia()
```

```
halo_dunia()
```

Output:

```
Halo python! Halo dunia
```

```
Halo python! Halo dunia
```

```
Halo python! Halo dunia
```

### Fungsi dengan Argumen atau Parameter

Sebuah fungsi juga bisa menerima parameter atau pun argumen. Ia merupakan suatu nilai/variabel yang dilemparkan ke dalam fungsi untuk diproses lebih lanjut.

Sebagai contoh, perhatikan **output** berikut:

```
Halo Rahma, selamat datang!
```

```
Halo Fajri, selamat datang!
```

```
Halo Hanifah, selamat datang!
```

Halo Rido, selamat datang!

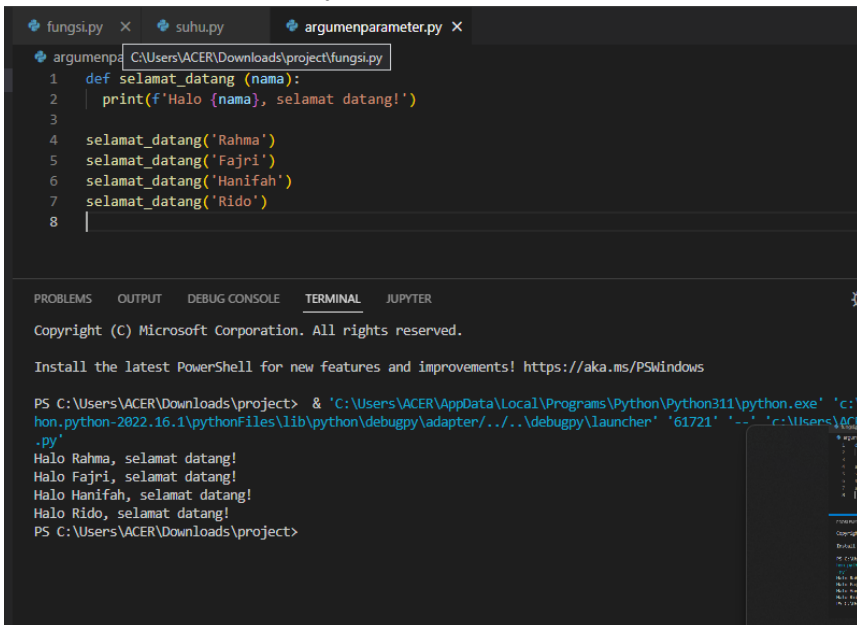
Ada banyak cara untuk memproduksi output dengan python. Bisa dengan list, perulangan, dan lain sebagainya.

Akan tetapi, mungkin yang langsung terbesit dalam benak kita adalah dengan melakukan 4x print() seperti ini:

```
print('Halo Rahma, selamat datang!')
print('Halo Fajri, selamat datang!')
print('Halo Hanifah, selamat datang!')
print('Halo Rido, selamat datang!')
```

Itu adalah cara yang sangat simpel, dan juga tidak salah.

Akan tetapi, dari pada kita melakukan 4x print seperti di atas, kita bisa memanfaatkan fungsi dan parameter pada python. Sehingga kode programnya akan terlihat seperti ini:



The screenshot shows a code editor with three tabs: 'fungsi.py', 'suhu.py', and 'argumenparameter.py'. The active tab is 'fungsi.py', which contains the following Python code:

```
1 def selamat_datang (nama):
2     print(f'Halo {nama}, selamat datang!')
3
4 selamat_datang('Rahma')
5 selamat_datang('Fajri')
6 selamat_datang('Hanifah')
7 selamat_datang('Rido')
8
```

Below the code editor is a terminal window. The terminal output shows the execution of the Python script, resulting in four lines of output:

```
PS C:\Users\ACER\Downloads\project> & 'C:\Users\ACER\AppData\Local\Programs\Python\Python311\python.exe' 'c:\hon.python-2022.16.1\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '61721' '--' 'c:\Users\ACER\Downloads\project'
Halo Rahma, selamat datang!
Halo Fajri, selamat datang!
Halo Hanifah, selamat datang!
Halo Rido, selamat datang!
PS C:\Users\ACER\Downloads\project>
```

**Gambar 32. Fungsi Pada Python**

```
def selamat_datang (nama):
    print(f'Halo {nama}, selamat datang!')
```

```
selamat_datang('Rahma')
selamat_datang('Fajri')
selamat_datang('Hanifah')
selamat_datang('Rido')
```

Dan kita tetap akan mendapatkan output yang sama.

### Parameter Wajib

Parameter di dalam python bisa lebih dari satu, bisa wajib semua (harus diisi), dan bisa juga bersifat opsional.

Perhatikan contoh fungsi berikut:

```
def perkenalan (nama, asal):
    print(f"Perkenalkan saya {nama} dari {asal}")
```

### Jika dipanggil:

```
perkenalan("Rahma Yanti", "Pasaman")
```

### Kita akan mendapatkan output:

```
Perkenalkan saya Rahma Yanti dari Pasaman
```

Tapi jika kita memanggilnya dengan parameter tidak lengkap, justru kita akan mendapatkan error:

```
perkenalan("Rahma Yanti")
```

### Error:

```
Exception has occurred: TypeError
perkenalan() missing 1 required positional
argument: 'asal'
```

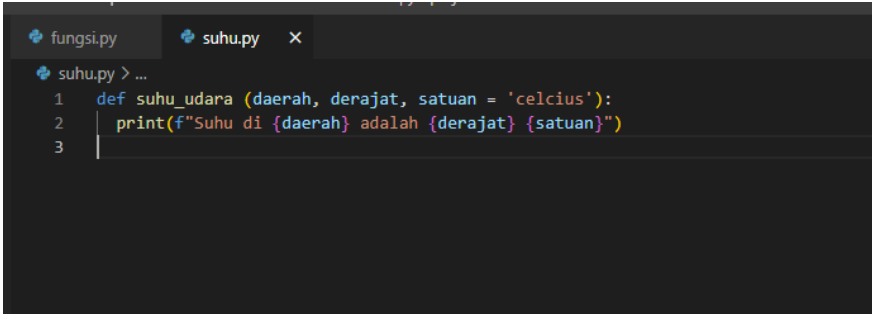
Kenapa? Karena kita hanya memasukkan satu parameter saja padahal parameter yang diminta ada 2.



## Parameter Opsional

Tidak semua parameter fungsi pada python itu bersifat wajib. Ada juga yang opsional. Parameter opsional merupakan parameter yang seandainya tidak diisi, dia sudah memiliki nilai default.

Contoh :



```
fungsi.py | suhu.py x
suhu.py > ...
1 def suhu_udara (daerah, derajat, satuan = 'celcius'):
2     print(f"Suhu di {daerah} adalah {derajat} {satuan}")
3
```

**Gambar 33. Parameter Operasional**

```
def suhu_udara (daerah, derajat, satuan =
'celcius'):
    print(f"Suhu di {daerah} adalah {derajat}
{satuan}")
```

Pada fungsi suhu\_udara() di atas, kita mendefinisikan 3 buah parameter:

```
daerah
derajat
suhu = 'celcius'
```

Dua parameter pertama adalah bersifat wajib dan harus diisi, sedangkan parameter ketiga tidak wajib. Jika tidak kita isi, maka nilai default-nya adalah "celcius".

Sekarang, kita coba panggil fungsi tersebut dengan 2 cara:

```
suhu_udara("Surabaya", 30)
suhu_udara("Surabaya", 86, 'Fahrenheit')
```

Jika dijalankan, outputnya akan terlihat seperti ini:

Suhu di Surabaya adalah 30 celcius

Suhu di Surabaya adalah 86 Fahrenheit

### **Fungsi dengan Parameter Tidak Berurut**

Jika kita perhatikan lagi fungsi `suhu_udara()`, kita akan dapati kalau parameter yang bersifat opsional hanya ada 1, dan hanya ada di belakang.

Tapi, bagaimana jika ternyata parameter opsionalnya ada lebih dari 1?

Coba perhatikan:

```
def suhu_udara (daerah, derajat = 30, satuan
= 'celcius'):
    print(f"Suhu di {daerah} adalah {derajat}
{satuan}")
```

Pada fungsi tersebut, kita telah mengatur nilai default untuk parameter derajat. Sehingga sekarang kita memiliki dua buah parameter.

Kita coba panggil dengan 2 parameter seperti ini:

```
suhu_udara('Jakarta', 'fahrenheit')
```

#### **Output :**

Suhu di Jakarta adalah fahrenheit celcius

### **Fungsi Mengembalikan Nilai**

Jenis fungsi yang berikutnya adalah berkaitan dengan nilai kembalian.

Ditinjau dari segi pengembalian nilai, fungsi terbagi menjadi 2:

- a) Fungsi yang tidak mengembalikan nilai
- b) Fungsi yang mengembalikan nilai

Pada contoh-contoh di atas, kita telah membuat dan memanggil fungsi-fungsi yang tidak memiliki nilai.

Sekarang, kita akan coba membuat fungsi yang mempunyai atau mengembalikan sebuah nilai.

```
def luas_persegi (sisi):  
    return sisi * sisi
```

Penjelasan :

Kata kunci `return` berfungsi untuk mengembalikan nilai.

Nilai yang dikembalikan suatu fungsi, bisa kita olah kembali untuk berbagai kebutuhan.

**Contoh :**

```
# tidak menghasilkan output apa pun  
luas_persegi(10)  
  
# menghasilkan output  
print('Luas persegi dengan sisi 4 adalah:',  
luas_persegi(4))  
  
# kita juga bisa simpan di dalam variabel  
persegi_besar = luas_persegi(100)  
persegi_kecil = luas_persegi(50)  
  
print('Toal luas persegi besar dan kecil  
adalah:', persegi_besar + persegi_kecil)
```

Jika dijalankan, kita akan mendapatkan output:

Luas persegi dengan sisi 4 adalah: 16

Toal luas persegi besar dan kecil adalah:  
12500

Jadi intinya: fungsi yang mengembalikan nilai adalah sebuah fungsi yang jika kita panggil, dia akan memberikan kita sebuah

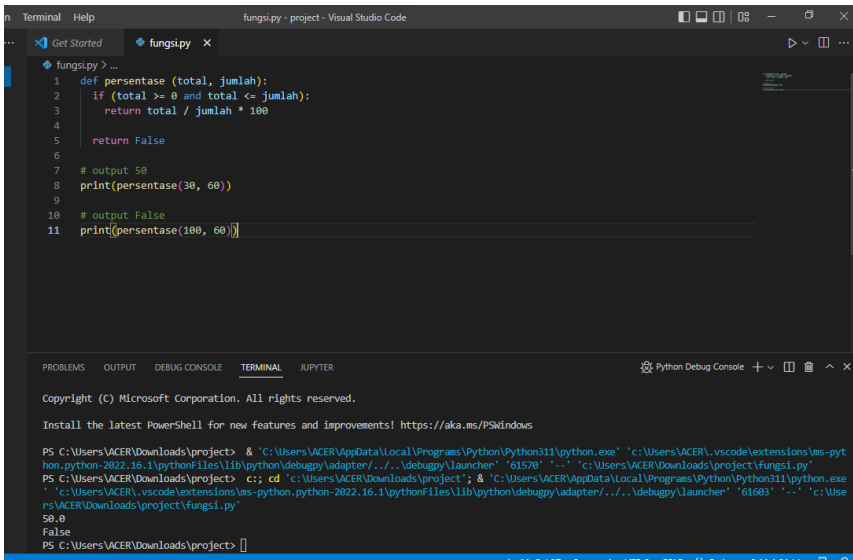
nilai yang bisa kita olah lebih lanjut, seperti misalkan kita simpan dalam sebuah variabel atau kita lakukan operasi tertentu.

### Lebih dari satu return

Jika statement return telah dieksekusi pada sebuah fungsi, maka semua proses yang ada di dalam blok kode fungsi tersebut akan berhenti.

Sehingga, misalkan kita memiliki lebih dari 1 buah return, maka hanya ada satu return saja yang dieksekusi. Dan ketika sebuah return telah dieksekusi, semua perintah yang ada di bawahnya akan di-skip –ini mirip dengan perintah break pada perulangan for mau pun while.

Perhatikan contoh berikut:



```
def persentase (total, jumlah):
    if (total >= 0 and total <= jumlah):
        return total / jumlah * 100
    return False

# output 50
print(persentase(30, 60))

# output False
print(persentase(100, 60))
```

Gambar 34. Lebih Dari Satu Return

```
def persentase (total, jumlah):
    if (total >= 0 and total <= jumlah):
        return total / jumlah * 100
    return False
```

```
# output 50
print(persentase(30, 60))
# output False

print(persentase(100, 60))
Output:
50.0
False
```

### Ruang Lingkup Variabel Pada Fungsi

Variabel memiliki ruang lingkup dan siklus hidup.

Secara umum, terdapat dua ruang lingkup variabel pada python:

- a) Variabel global
- b) Variabel lokal

Variable global adalah variabel yang bisa dipanggil dari manapun dari satu file python.

Sedangkan variable lokal adalah variabel yang hanya hidup di dalam satu blok kode tertentu (seperti di dalam fungsi, seperti kasus kita pada pertemuan ini).

Perhatikan contoh berikut:

The screenshot shows a Jupyter Notebook interface with a terminal window. The code in the notebook is as follows:

```
1 kota = 'Bukittinggi'
2
3 def halo() :
4     print(kota)
5
6 print('[print secara langsung]', kota)
7 print('[panggil fungsi halo]', end= ' ')
8
9 halo()
```

The terminal output shows the execution of the code:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

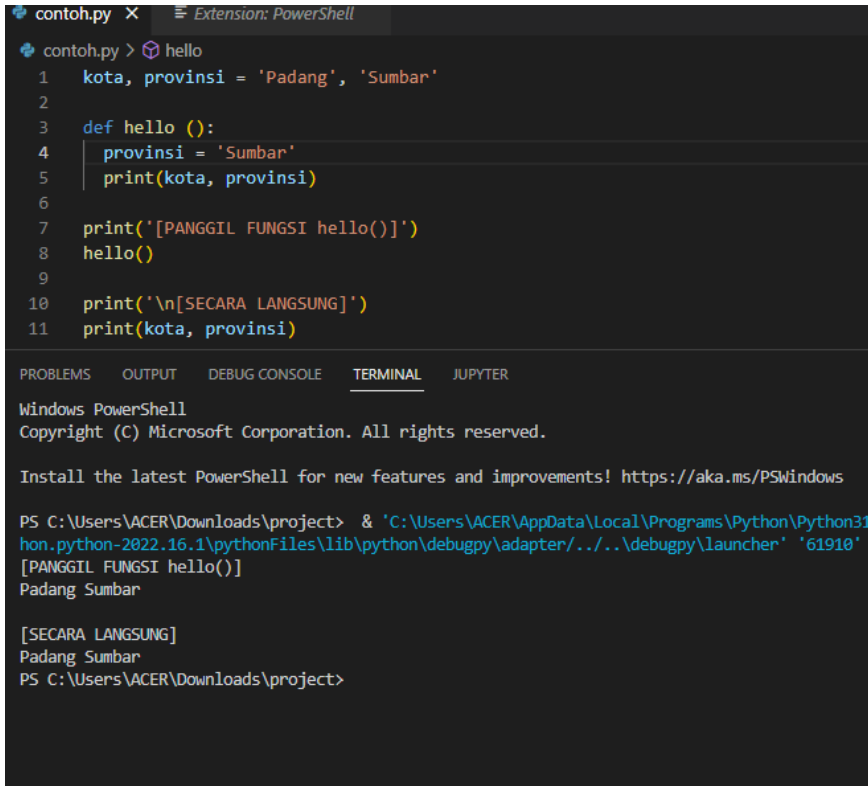
PS C:\Users\VACER\Downloads\project> & 'C:\Users\VACER\AppData\Local\Programs\Python\Python311\python.exe'
hon.python-2022.16.1\pythonFiles\Lib\python\debugpy\launcher' '61884' '--' 'c:\Users
[print secara langsung] Bukittinggi
[panggil fungsi halo] Bukittinggi
PS C:\Users\VACER\Downloads\project>
```

**Gambar 35. Ruang Lingkup Variabel dan Fungsi**

Output :

```
[print secara langsung] Bukittinggi  
[panggil fungsi halo] Bukittinggi
```

Pada kode di atas, variabel kota yang ada di dalam fungsi, adalah variabel kota yang sama dengan yang ada di luar fungsi. Tapi, coba kita ubah kode programnya:



```
contoh.py x  Extension: PowerShell  
contoh.py > hello  
1  kota, provinsi = 'Padang', 'Sumbar'  
2  
3  def hello ():  
4      provinsi = 'Sumbar'  
5      print(kota, provinsi)  
6  
7  print('[PANGGIL FUNGSI hello()])  
8  hello()  
9  
10 print('\n[SECARA LANGSUNG]')  
11 print(kota, provinsi)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

Windows PowerShell  
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! <https://aka.ms/PSWindows>

PS C:\Users\VACER\Downloads\project> & 'C:\Users\VACER\AppData\Local\Programs\Python\Python311\python-2022.16.1\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '61910'  
[PANGGIL FUNGSI hello()]  
Padang Sumbar

[SECARA LANGSUNG]  
Padang Sumbar  
PS C:\Users\VACER\Downloads\project>

**Gambar 36. Output Ruang Lingkup Variabel dan Fungsi**

```
kota, provinsi = 'Padang', 'Sumbar'
```

```
def hello ():  
    provinsi = 'Sumbar'  
    print(kota, provinsi)
```

```
print('[PANGGIL FUNGSI hello()])'  
hello()  
print('\n[SECARA LANGSUNG]')  
print(kota, provinsi)
```

## Kelas dan Objek

### Semua Hal Pada Python adalah Objek

Sebelumnya, kita perlu tahu dulu bahwa sesuatu tidaklah dikatakan objek kecuali jika memiliki atribut atau perilaku.

Atribut adalah semacam identitas atau variabel dari suatu objek, sedangkan perilaku adalah “kemampuan” atau fitur dari objek tersebut.

Kita juga bisa mendefinisikan dalam bentuk yang lebih sederhana lagi, yaitu: objek adalah sebuah gabungan dari kumpulan variabel (dinamakan atribut) dan kumpulan fungsi (dinamakan perilaku).

Dan atas definisi itu, maka bisa dikatakan bahwa semua hal di dalam python adalah sebuah objek .

### Bahkan Sebuah Fungsi Pun Adalah Objek

Kalau kita teliti lebih jauh lagi, setiap fungsi pada python memiliki atribut `__doc__` yang merupakan bukti bahwa fungsi sekalipun adalah sebuah objek dalam python.

Jika kita periksa tipe data dari sebuah fungsi, kita akan mendapati bahwa ia ternyata objek dari sebuah class dengan nama `function`:

```
>>> def tes():  
...     print('halo')  
...  
>>> type(tes)  
<class 'function'>
```

## Kelas Pada Python

Kelas atau class pada python bisa kita katakan sebagai sebuah blueprint (cetakan) dari objek (atau instance) yang ingin kita buat.

Kelas adalah cetakannya atau definisinya, sedangkan objek (atau instance) adalah objek nyatanya.

Kita coba beri contoh “kucing” untuk memperdekat pemahaman.

- a) “Kucing” merupakan sebuah definisi objek, ia memiliki 2 telinga, 4 kaki, 1 ekor, nama dan lain-lain (sebagai atribut), ia juga bisa berlari, mengeong, makan dan minum (sebagai perilaku).
- b) Misal kita memiliki 4 kucing: berarti kita memiliki “4 instance” dari kelas kucing.
- c) Masing-masing dari 4 kucing tersebut memiliki atribut yang telah didefinisikan sebelumnya seperti kaki, telinga, ekor, dan lain-lain. Ia juga memiliki kemampuan berlari, mengeong, dan sebagainya seperti yang telah didefinisikan sebelumnya.

## Atribut dan Prilaku

**Atribut** pada OOP merepresentasikan variabel yang dimiliki sebuah objek.

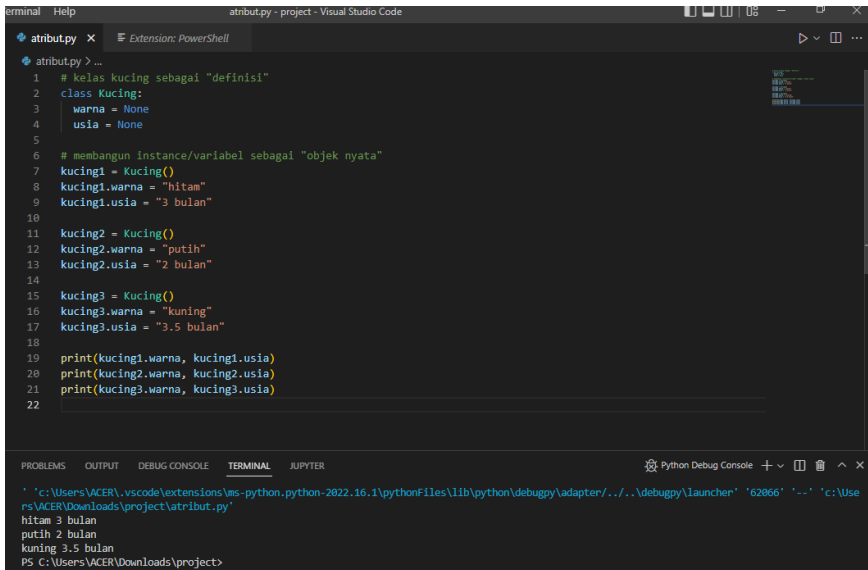
Sedangkan **perilaku** pada OOP merepresentasikan fungsi yang dimiliki sebuah objek.

Contoh Atribut

Sebagai contoh, kita memiliki 3 ekor kucing, masing-masing kucing memiliki warna dan usianya sendiri-sendiri.

Jika kita ingin merepresentasikan 3 ekor kucing tersebut dengan pendekatan OOP, kita bisa menuliskan kode programnya seperti berikut:





The image shows a screenshot of the Visual Studio Code editor. The main window displays a Python file named 'atribut.py' with the following code:

```
1 # kelas kucing sebagai "definisi"
2 class Kucing:
3     warna = None
4     usia = None
5
6 # membangun instance/variabel sebagai "objek nyata"
7 kucing1 = Kucing()
8 kucing1.warna = "hitam"
9 kucing1.usia = "3 bulan"
10
11 kucing2 = Kucing()
12 kucing2.warna = "putih"
13 kucing2.usia = "2 bulan"
14
15 kucing3 = Kucing()
16 kucing3.warna = "kuning"
17 kucing3.usia = "3.5 bulan"
18
19 print(kucing1.warna, kucing1.usia)
20 print(kucing2.warna, kucing2.usia)
21 print(kucing3.warna, kucing3.usia)
22
```

The bottom panel shows the terminal output:

```
* c:\Users\VACER\.vscode\extensions\ms-python.python-2022.16.1\pythonFiles\lib\python\debuggy\adapter\...
rs\VACER\Downloads\project\atribut.py
hitam 3 bulan
putih 2 bulan
kuning 3.5 bulan
PS c:\Users\VACER\Downloads\project>
```

**Gambar 37. Atribut Pada Python**

```
# kelas kucing sebagai "definisi"
```

```
class Kucing:
    warna = None
    usia = None
```

```
# membangun instance/variabel sebagai "objek nyata"
```

```
kucing1 = Kucing()
kucing1.warna = "hitam"
kucing1.usia = "3 bulan"
```

```
kucing2 = Kucing()
kucing2.warna = "putih"
kucing2.usia = "2 bulan"
```

```

kucing3 = Kucing()
kucing3.warna = "kuning"
kucing3.usia = "3.5 bulan"
Kita bisa menampilkan masing-masing atribut dari tiap instance:
print(kucing1.warna, kucing1.usia)
print(kucing2.warna, kucing2.usia)
print(kucing3.warna, kucing3.usia)

```

### Contoh Perilaku

Misalkan kita memiliki kelas Mahasiswa. Setiap mahasiswa memiliki atribut:

- a) Nama
- b) dan Asal

Serta memiliki perilaku:

- a) memperkenalkan diri

Maka bisa kita definisikan kode programnya dengan pendekatan OOP sebagai berikut:

```

# buat kelasnya terlebih dahulu
class Mahasiswa:
    nama = None
    asal = None

    def perkenalan (self):
        print(f'Perkenalkan saya {self.nama}
dari {self.asal}')

```

### Pewarisan Pada Python

Konsep pewarisan adalah konsep di mana sebuah kelas atau objek mewariskan sifat dan perilaku kepada kelas lainnya.

Kelas yang menjadi “pemberi waris” dinamakan kelas induk atau kelas basis

Sedangkan kelas yang menjadi “ahli waris” dinamakan sebagai kelas turunan.

### **Sifat Pewarisan**

Secara umum, kelas turunan akan selalu memiliki sifat dan perilaku yang sama dengan kelas induknya: mulai dari atribut sampai fungsi-fungsinya.

Akan tetapi tidak sebaliknya, belum tentu kelas induk memiliki semua atribut dan sifat dari kelas-kelas turunannya.

### **Contoh Sederhana Penerapan Pewarisan Objek**

Untuk mempermudah pemahaman, mari kita buat contoh kasus sederhana yang akan kita selesaikan dengan konsep pewarisan.

Kita akan membuat 3 buah objek:

- Orang
- Pelajar
- Pekerja

Masing-masing dari 3 objek tersebut memiliki beberapa sifat dan perilaku yang sama, misalkan:

- nama
- asal
- dan kemampuan memperkenalkan diri

Tetapi, sebagian objek tetap memiliki ciri khasnya masing-masing, misalkan:

- Objek Pelajar memiliki atribut sekolah.
- Objek Pekerja memiliki atribut tempat kerja

### **Hubungan “pewarisan” antar ketiganya**

Untuk menyelesaikan kasus di atas, kita perlu menarik sebuah premis bahwa ketiganya memiliki hubungan “waris”.

Hubungannya adalah:

- Objek Pelajar sebenarnya adalah objek Orang juga, hanya saja objek Pelajar memiliki atribut tambahan yang tidak dimiliki objek Orang.
- Begitu pula objek Pekerja, ia sebenarnya adalah objek Orang juga, hanya saja ia memiliki atribut tambahan yang tidak dimiliki objek Orang.

### Definisikan Kelas Basis dan Kelas Turunan

Setelah mengetahui gambaran hubungan “waris” antar ketiga kelas tersebut, kita bisa simpulkan bahwa:

- Objek Pelajar dan objek Pekerja adalah kelas turunan dari objek Orang.

### Membuat Objek Induk (Parent)

Untuk membuat objek induk pada python, caranya sama dengan cara membuat objek biasa. Karena pada dasarnya, semua objek pada python bisa menjadi objek induk dari objek turunan lainnya.

Mari kita buat objek Orang seperti skenario yang telah kita bahas pada bagian sebelumnya:

```
class Orang:
    def __init__(self, nama, asal):
        self.nama = nama
        self.asal = asal
    def perkenalan(self):
        print(f'Perkenalkan      nama      saya
        {self.nama} dari {self.asal}')
```

Objek di atas sangat sederhana, ia hanya memiliki 2 atribut (nama dan asal) serta memiliki satu buah perilaku yaitu perkenalan().

Kita bisa membuat instance dari kelas Orang seperti biasanya:

```
andi = Orang('Andi', 'Surabaya')
andi.perkenalan()
```

### Membuat Objek Turunan

Langkah selanjutnya adalah: membuat objek turunan dari kelas Orang. Sesuai dengan skenario yang telah kita bahas di atas, kita akan membuat dua buah objek baru yaitu objek Pelajar dan Pekerja, yang mana keduanya akan mewarisi objek Orang.

Contoh ;

```
class Pelajar (Orang):
    pass
class Pekerja (Orang):
    pass
```

Kita telah membuat dua buah kelas yang keduanya sama-sama memiliki setiap atribut dan fungsi dari kelas Orang.

Kita meletakkan perintah `pass` karena kita hanya ingin melakukan pewarisan apa adanya tanpa menambahkan apa pun lagi.

Sehingga, kita bisa membuat instance dari kelas Pelajar dan Pekerjaan, serta memanggil fungsi `perkenalan()` dengan cara yang benar-benar identik dengan kelas Orang:

```
andi = Orang('Andi', 'Surabaya')
andi.perkenalan()

deni = Pelajar('Deni', 'Makassar')
deni.perkenalan()

budi = Pekerja('Budi', 'Pontianak')
budi.perkenalan()
```

## Output :

Perkenalkan nama saya Andi dari Surabaya  
Perkenalkan nama saya Deni dari Makassar  
Perkenalkan nama saya Budi dari Pontianak

## Polimorfisme Pada Phyton

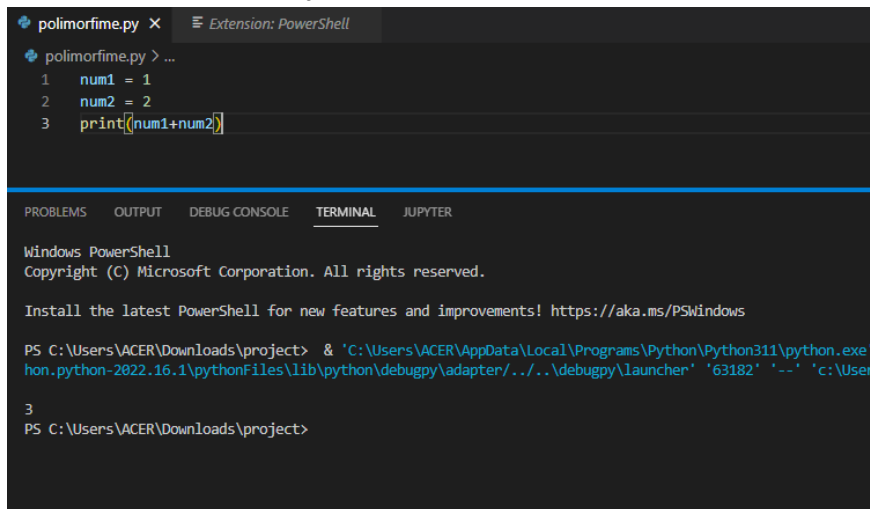
Arti harfiah polimorfisme adalah kondisi terjadinya dalam berbagai bentuk.

Polimorfisme adalah konsep yang sangat penting dalam pemrograman. Ini mengacu pada penggunaan entitas tipe tunggal (metode, operator atau objek) untuk mewakili tipe yang berbeda dalam skenario yang berbeda.

Contoh 1: Polimorfisme sebagai operator penjumlahan

Kita tahu bahwa +operator digunakan secara luas dalam program Python. Tapi, itu tidak memiliki penggunaan tunggal.

Untuk tipe data integer, +operator digunakan untuk melakukan operasi penjumlahan aritmatika.



```
polimorfime.py X  Extension: PowerShell
polimorfime.py > ...
1 num1 = 1
2 num2 = 2
3 print(num1+num2)

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  JUPYTER

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\ACER\Downloads\project> & 'C:\Users\ACER\AppData\Local\Programs\Python\Python311\python.exe'
hon.python-2022.16.1\pythonFiles\lib\python\debugpy\adapter\..\debugpy\launcher' '63182' '--' 'c:\User
3
PS C:\Users\ACER\Downloads\project>
```

**Gambar 38. Polimerfime Pada Python**

```
num1 = 1
num2 = 2
print(num1+num2)
```



```

print(len("Programiz"))
print(len(["Python", "Java", "C"]))
print(len({"Name": "John", "Address":
"Nepal"}))

```

## Polimorfisme Kelas dalam Python

Polimorfisme adalah konsep yang sangat penting dalam Pemrograman Berorientasi Objek.

Kita dapat menggunakan konsep polimorfisme sambil membuat metode kelas karena Python memungkinkan kelas yang berbeda memiliki metode dengan nama yang sama.

Kami kemudian dapat menggeneralisasi pemanggilan metode ini dengan mengabaikan objek yang sedang kami kerjakan. Mari kita lihat sebuah contoh:

### Contoh 3: Polimorfisme dalam Metode Kelas

```

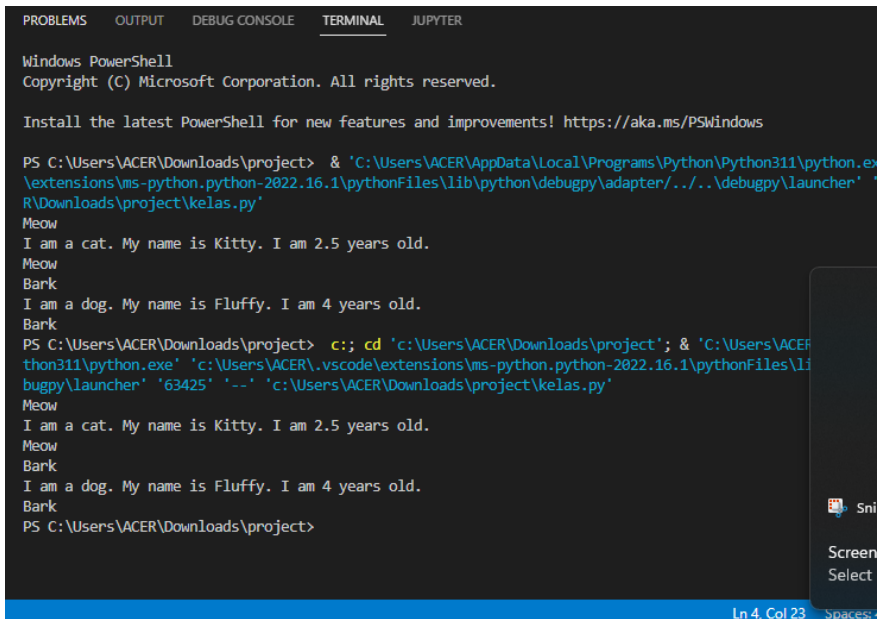
kelas.py > Cat > _init_
2 def __init__(self, name, age):
3     self.name = name
4     self.age = age
5
6 def info(self):
7     print(f"I am a cat. My name is {self.name}. I am {self.age} years old.")
8
9 def make_sound(self):
10    print("Meow")
11
12
13 class Dog:
14    def __init__(self, name, age):
15        self.name = name
16        self.age = age
17
18    def info(self):
19        print(f"I am a dog. My name is {self.name}. I am {self.age} years old.")
20
21    def make_sound(self):
22        print("Bark")
23
24
25 cat1 = Cat("Kitty", 2.5)
26 dog1 = Dog("Fluffy", 4)
27
28 for animal in (cat1, dog1):
29    animal.make_sound()
30    animal.info()
31    animal.make_sound()

```

**Gambar 40. Polimerfime Kelas Pada Python**



## Output :



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\VACER\Downloads\project> & 'C:\Users\VACER\AppData\Local\Programs\Python\Python311\python.exe'
\extensions\ms-python.python-2022.16.1\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher'
R\Downloads\project\kelas.py'
Meow
I am a cat. My name is Kitty. I am 2.5 years old.
Meow
Bark
I am a dog. My name is Fluffy. I am 4 years old.
Bark
PS C:\Users\VACER\Downloads\project> c; cd 'c:\Users\VACER\Downloads\project'; & 'C:\Users\VACER\
thon311\python.exe' 'c:\Users\VACER\.vscode\extensions\ms-python.python-2022.16.1\pythonFiles\li
bugpy\launcher' '63425' '--' 'c:\Users\VACER\Downloads\project\kelas.py'
Meow
I am a cat. My name is Kitty. I am 2.5 years old.
Meow
Bark
I am a dog. My name is Fluffy. I am 4 years old.
Bark
PS C:\Users\VACER\Downloads\project>
```

**Gambar 41. Ouput Polimerfime Kelas Pada Python**

Code:

```
class Cat:
```

```
    def __init__(self, name, age):
```

```
        self.name = name
```

```
        self.age = age
```

```
    def info(self):
```

```
        print(f"I am a cat. My name is {self.name}. I am {self.age} years old.")
```

```
    def make_sound(self):
```

```
        print("Meow")
```

```
class Dog:
```

```
    def __init__(self, name, age):
```

```
        self.name = name
```

```
        self.age = age
```

```

def info(self):
    print(f"I am a dog. My name is
{self.name}. I am {self.age} years old.")
def make_sound(self):
    print("Bark")
cat1 = Cat("Kitty", 2.5)
dog1 = Dog("Fluffy", 4)
for animal in (cat1, dog1):
    animal.make_sound()
    animal.info()
    animal.make_sound()

```

### **Polimorfisme dan Warisan**

Seperti bahasa pemrograman lainnya, kelas anak di Python juga mewarisi metode dan atribut dari kelas induk. Kita dapat mendefinisikan kembali metode dan atribut tertentu secara khusus agar sesuai dengan kelas anak, yang dikenal sebagai Method Overriding .

Polimorfisme memungkinkan kita untuk mengakses metode dan atribut yang diganti ini yang memiliki nama yang sama dengan kelas induknya.

Mari kita lihat sebuah contoh:

### **Contoh 4: Penggantian Metode**

```
polimeren.py  kelas.py  polimer.py x
polimer.py > Square > fact
1  from math import pi
2  class Shape:
3      def __init__(self, name):
4          self.name = name
5
6      def area(self):
7          pass
8
9      def fact(self):
10         return "I am a two-dimensional shape."
11
12     def __str__(self):
13         return self.name
14 class Square(Shape):
15     def __init__(self, length):
16         super().__init__("Square")
17         self.length = length
18
19     def area(self):
20         return self.length**2
21
22     def fact(self):
23         return "Squares have each angle equal to 90 degrees."
24 class Circle(Shape):
25     def __init__(self, radius):
26         super().__init__("Circle")
27         self.radius = radius
28
29     def area(self):
30         return pi*self.radius**2
31
32
```

**Gambar 42. Penggantian Metode**

```
from math import pi
class Shape:
    def __init__(self, name):
        self.name = name

    def area(self):
        pass
    def fact(self):
        return "I am a two-dimensional
shape."
    def __str__(self):
        return self.name
class Square(Shape):
    def __init__(self, length):
```

```

        super().__init__("Square")
        self.length = length

    def area(self):
        return self.length**2

    def fact(self):
        return "Squares have each angle
equal to 90 degrees."
class Circle(Shape):
    def __init__(self, radius):
        super().__init__("Circle")
        self.radius = radius

    def area(self):
        return pi*self.radius**2

a = Square(4)
b = Circle(7)
print(b)
print(b.fact())
print(a.fact())
print(b.area())

```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

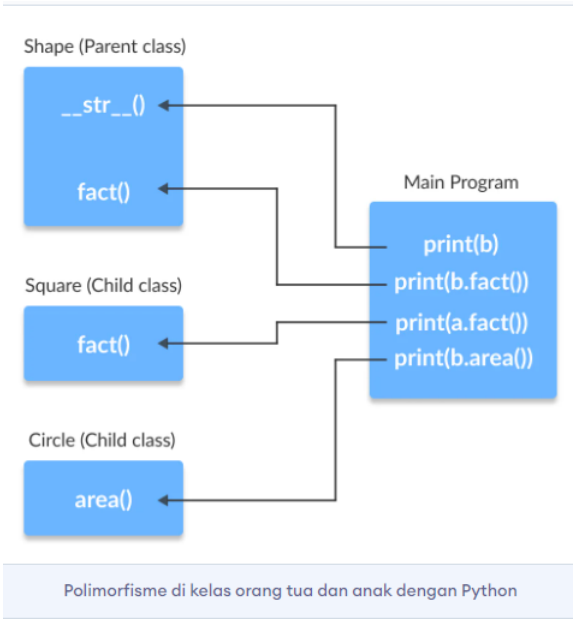
Circle
I am a two-dimensional shape.
Squares have each angle equal to 90 degrees.
153.93804002589985
PS C:\Users\ACER\Downloads\project>
```

**Gambar 43. Ouput Penggantian Metode**

Di sini, kita dapat melihat bahwa metode seperti `__str__()`, yang belum ditimpa di kelas anak, digunakan dari kelas induk.

Karena polimorfisme, juru bahasa Python secara otomatis mengenali bahwa `fact()` metode untuk objek `a` (Persegikelas) diganti. Jadi, ini menggunakan yang didefinisikan di kelas anak.

Di sisi lain, sejak `fact()` metode untuk objek `B` tidak diganti, ini digunakan dari Induk Membentuk kelas.



**Gambar 44. Logika Polimorfime**

Method Overloading , cara untuk membuat banyak metode dengan nama yang sama tetapi argumen yang berbeda.

## Penanganan Eksepsi

Eksepsi (exceptions) dalam Python adalah kondisi atau situasi yang tidak diharapkan yang terjadi saat program sedang dieksekusi. Ketika eksepsi terjadi, aliran normal program terputus, dan Python mencoba mencari penanganan (handling) eksepsi yang sesuai. Jika penanganan eksepsi ditemukan, program akan melompat ke blok kode yang ditentukan untuk menangani eksepsi tersebut. Jika penanganan tidak ditemukan, program akan menghentikan eksekusi dan menampilkan pesan kesalahan.

Berikut adalah format umum dari penanganan eksepsi dengan blok try-except:

```
try:
    # Blok kode yang mungkin menyebabkan eksepsi
    pernyataan_1
    pernyataan_2
    ...
except JenisEksepsi:
    # Blok kode untuk menangani eksepsi
    pernyataan_penanganan_eksepsi
```

Dalam blok try, kita menempatkan blok kode yang berpotensi menyebabkan eksepsi. Jika eksepsi terjadi saat eksekusi blok try, Python akan mencari blok except yang sesuai dengan jenis eksepsi yang dihasilkan. Jika eksepsi yang dihasilkan cocok dengan jenis eksepsi yang ditentukan dalam blok except, blok kode penanganan eksepsi akan dieksekusi.

Contoh program try-except:

```
try:
    x = 10 / 0
except ZeroDivisionError:
    print("Terjadi pembagian dengan nol!")
```

Output Program:

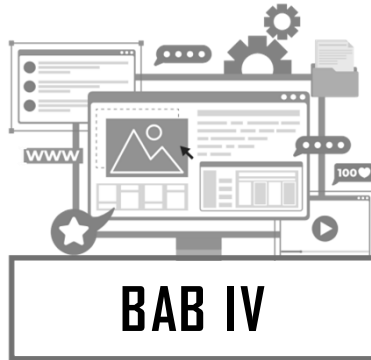
---

Terjadi pembagian dengan nol!

---

Pada contoh di atas, eksepsi `ZeroDivisionError` terjadi ketika mencoba membagi angka 10 dengan 0. Blok kode dalam blok `try` menghasilkan eksepsi tersebut, dan karena jenis eksepsi cocok dengan jenis yang ditentukan dalam blok `except`, blok kode penanganan eksepsi akan dieksekusi.

Selain blok `except`, kita juga dapat menggunakan blok `else` dan `finally` untuk memberikan logika tambahan dalam penanganan eksepsi. Blok `else` akan dieksekusi jika tidak ada eksepsi yang dihasilkan dalam blok `try`, sedangkan blok `finally` akan dieksekusi baik terjadi eksepsi atau tidak.



## HELLO DJANGO

### Sejarah Django

Django merupakan sebuah kerangka kerja atau disebut framework web gratis dan juga open source berbasis Python, yang mengikuti pola arsitektur model-template-view. Django diciptakan oleh Adrian Holovaty dan Simon Willison. Django dikembangkan dan dikelola oleh Django Software Foundation. Dirilis awal 15 Juli 2005, dan ditulis memakai Bahasa Python. Django diciptakan, dikota Lawrence, Kansas, Amerika Serikat. Perayaan satu decade ulang tahun Django diselenggarakan dari tanggal 10 hingga 12 juli 2015. Adrian Holovaty sendiri, pencipta Django, akan tampil Bersama Douglas County Quintet dan Billy Ebeing.

### Pengertian Django

Django merupakan kerangka kerja full stack yang berfungsi untuk membuat aplikasi web menggunakan bahasa python. Sama halnya dengan flask, developer bisa membangun website secara backend maupun frontend hanya menggunakan framework ini. Framework python ini terkenal dengan performanya yang cepat dalam mengembangkan aplikasi dan memiliki desain pragmatis



yang lebih bersih. Sehingga jika Anda menggunakannya, maka proses pengembangan aplikasi menjadi lebih cepat dan tentunya menghemat kode. Pertama kali diluncurkan oleh Simon Willison dan Adrian Holovaty pada tahun 2003, Django membuat sebuah website berita. Kemudian namanya berasal dari gitaris Belgia dan Perancis yaitu Django Reinhardt. Pada bulan September 2008 rilis versi 1.0, lalu tahun 2018 sudah mencapai versi 2.0. Saat ini, banyak perusahaan besar yang mengimplementasikan framework python ini untuk mengembangkan aplikasi seperti Instagram, DropBox, Spotify, dan masih banyak lagi.

## Instalasi Django dan Setup Project

Pada tahap ini akan menunjukkan cara membuat aplikasi python dengan framework Django untuk pertama kali dengan beberapa Langkah.

### **Membuat Virtual Environment**

Sebelum kita memulai project baru kita, python merekomendasi untuk menggunakan **virtual environment** yang digunakan untuk membuat lingkungan python virtual yang terisolasi. ini untuk memastikan kalau versi dari sebuah library yang digunakan di satu project tidak akan berubah apabila kita melakukan sebuah update di library yang sama di project lainnya.

Berikut ini Langkah Langkah yang dengan menggunakan perintah di powershell:

- a) Jalankan Power shell
- b) Siapkan lokasi direktori kerja, misalnua dengan nama *project\_django*
- c) C:\Users\ACER> cd D:\
- d) D:\> mkdir project\_django
- e) Lalu masuk kedalam direktori kerja menggunakan perintah berikut
- f) D:\> cd .\project\_django\
- g) Buat virtual environment untuk project kita, dengan perintah D:\project\_django> python -m venv env

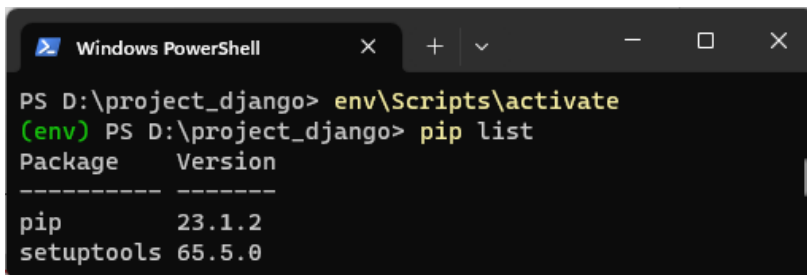
1. Lalu kita harus melakukan aktivasi dari venv yang sudah kita buat, dengan perintah

```
D:\project_django> env\Scripts\activate
```

Lalu kita dapat melihat list library yang terinstall pada virtual environment, dengan perintah

```
(env) PS D:\project_django> pip list
```

Dari perintah itu kita akan mendapatkan output seperti berikut:



```
Windows PowerShell
PS D:\project_django> env\Scripts\activate
(env) PS D:\project_django> pip list
Package      Version
-----
pip          23.1.2
setuptools  65.5.0
```

**Gambar 45. Membuat Virtual Environment**

Catatan :

Saat pertama sekali melakukan aktivasi, pengguna windows akan mendapatkan error sebagai berikut :

**.\env\Scripts\activate : File**

**D:\project\_django\env\Scripts\Activate.ps1 cannot be loaded because running scripts is disabled on this system. For more information, see about\_Execution\_Policies at <https://go.microsoft.com/fwlink/?LinkID=135170>.**

Perbaiki dengan cara masukkan perintah berikut pada powershell :

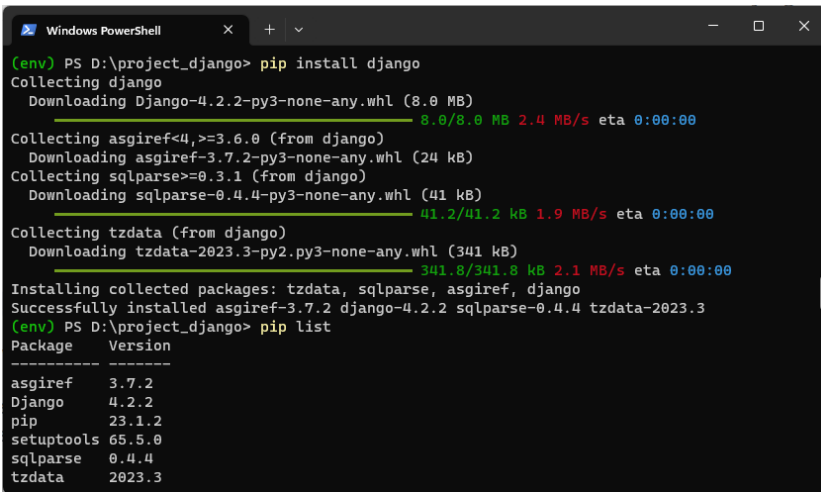
```
Set-ExecutionPolicy -Scope CurrentUser -
ExecutionPolicy Unrestricted -force
```

### ***Membuat Project Django***

Pada tahap ini akan membuat web pertama kita menggunakan framework Django dengan mudah menggunakan perintah baris.

Caranya adalah sebagai berikut:

1. Install library Django pada project kita dengan virtual environment yang aktif pada project kita, dengan perintah (env) PS D:\project\_django> pip install Django
2. Pastikan saat instalasi internet aktif, dan tunggu download dan instalasi django, lalu pastikan Django sudah terinstal dengan melihat library yang ada pada env kita, dengan perintah (env) PS D:\project\_django> pip list



```
Windows PowerShell
(env) PS D:\project_django> pip install django
Collecting django
  Downloading Django-4.2.2-py3-none-any.whl (8.0 MB)
    -----
    8.0/8.0 MB 2.4 MB/s eta 0:00:00
Collecting asgiref<4,>=3.6.0 (from django)
  Downloading asgiref-3.7.2-py3-none-any.whl (24 kB)
Collecting sqlparse>=0.3.1 (from django)
  Downloading sqlparse-0.4.4-py3-none-any.whl (41 kB)
    -----
    41.2/41.2 kB 1.9 MB/s eta 0:00:00
Collecting tzdata (from django)
  Downloading tzdata-2023.3-py2.py3-none-any.whl (341 kB)
    -----
    341.8/341.8 kB 2.1 MB/s eta 0:00:00
Installing collected packages: tzdata, sqlparse, asgiref, django
Successfully installed asgiref-3.7.2 django-4.2.2 sqlparse-0.4.4 tzdata-2023.3
(env) PS D:\project_django> pip list
Package      Version
-----
asgiref     3.7.2
Django      4.2.2
pip         23.1.2
setuptools  65.5.0
sqlparse    0.4.4
tzdata     2023.3
```

**Gambar 46. Output Virtual Environment**

Dari perintah diatas akan mendapatkan output

3. Lalu buat project Django dengan perintah (env) PS D:\project\_django> django-admin startproject webapp
4. Lalu jalankan project Django yang sudah dibuat tadi dengan perintah (env) PS D:\project\_django\webapp> cd .\webapp\  
(env) PS D:\project\_django\webapp> python .\manage.py runserver

Lalu powershell akan memberikan output sebagai berikut:

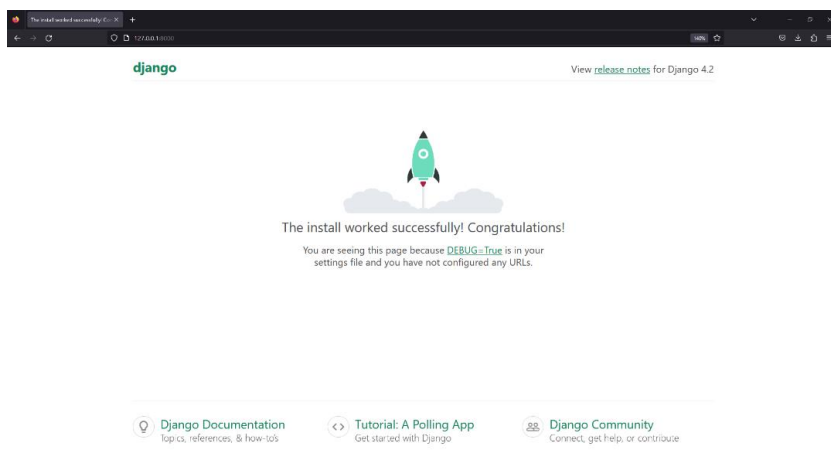
5. Buka browser anda, dan masukkan url <http://127.0.0.1:8000/>  
Hasil yang akan ditampilkan pada browser sebagi berikut :

```
Windows PowerShell
(env) PS D:\project_django\webapp> python .\manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the
migrations for app(s): admin, auth, contenttypes, sessions.
Run 'python manage.py migrate' to apply them.
June 24, 2023 - 02:09:05
Django version 4.2.2, using settings 'webapp.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

**Gambar 47. Powershell**



**Gambar 48. Memasukkan URL**

## Aplikasi Pertama Dengan Django

Hal yang pertama yang harus dilakukan untuk membuat aplikasi pertama kita dengan python menggunakan framework Django, berikut langkah langkah yang perlu di lakukan

1. Jalankan powershell/terminal
2. Masuk kedalam folder project Django yang sudah dibuat, dengan perintah  
cd D:\project\_django\

3. Lalu aktifkan ENV nya, dengan perintah

Pada Windows :

```
D:\project_django> env\Scripts\activate
```

Pada Linux dan Mac os:

```
$ source env\Scripts\activate
```

### Menambahkan *app* baru

Berikut langkah Langkah untuk menambhakan *app* baru pada website kita

1. Tambahkan app baru pada website django, dengan perintah:

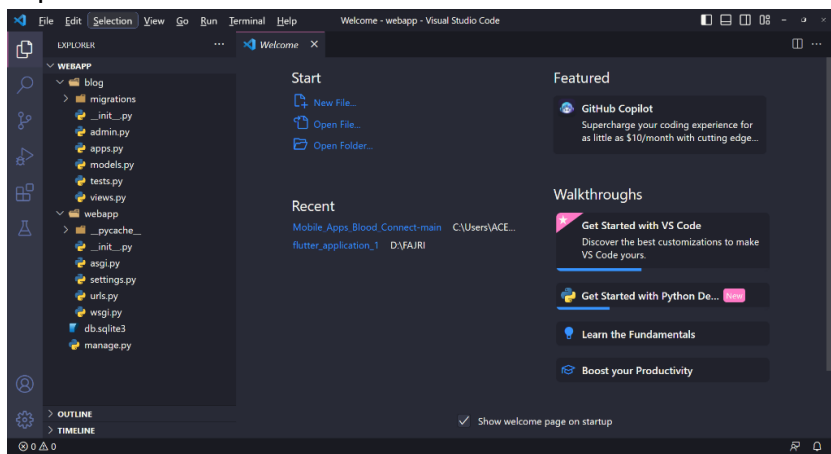
```
django-admin startapp blog
```

2. Lalu buka project dengan menggunakan Visual Studio Code, dengan perintah:

```
cd webapp
```

```
code .
```

visual studio akan secara otomatis terbuka dengan tampilan seperti berikut:



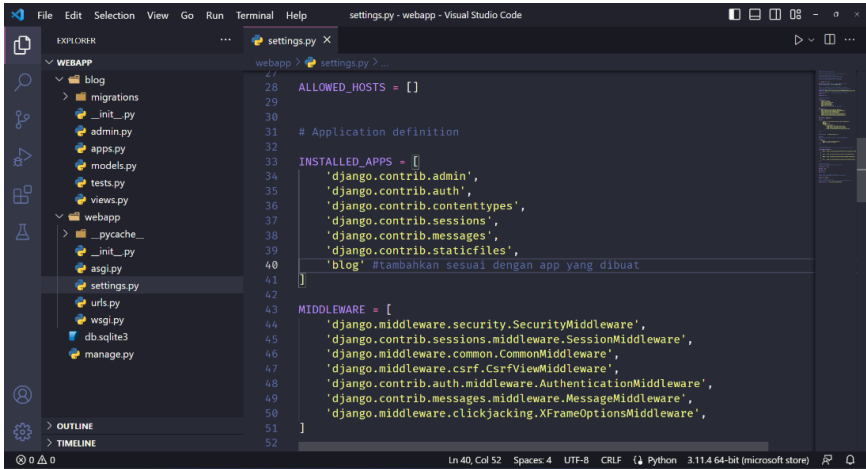
**Gambar 49. Menambahkan App Baru**

3. Lalu jalankan project django, dengan perintah:

```
python manage.py runserver
```

## Membuat website "Hello world"

Ini adalah cara untuk membuat website "Hello World" pertama buka file setting.py pada project webapp, lalu tambahkan elemen baru yang bernilai nama yang sesuai dengan app yang sudah dibuat pada list INSTALLED\_APPS, seperti berikut:



```
28 ALLOWED_HOSTS = []
29
30
31 # Application definition
32
33 INSTALLED_APPS = [
34     'django.contrib.admin',
35     'django.contrib.auth',
36     'django.contrib.contenttypes',
37     'django.contrib.sessions',
38     'django.contrib.messages',
39     'django.contrib.staticfiles',
40     'blog' #tambahkan sesuai dengan app yang dibuat
41 ]
42
43 MIDDLEWARE = [
44     'django.middleware.security.SecurityMiddleware',
45     'django.contrib.sessions.middleware.SessionMiddleware',
46     'django.middleware.common.CommonMiddleware',
47     'django.middleware.csrf.CsrfViewMiddleware',
48     'django.contrib.auth.middleware.AuthenticationMiddleware',
49     'django.contrib.messages.middleware.MessageMiddleware',
50     'django.middleware.clickjacking.XFrameOptionsMiddleware',
51 ]
52
```

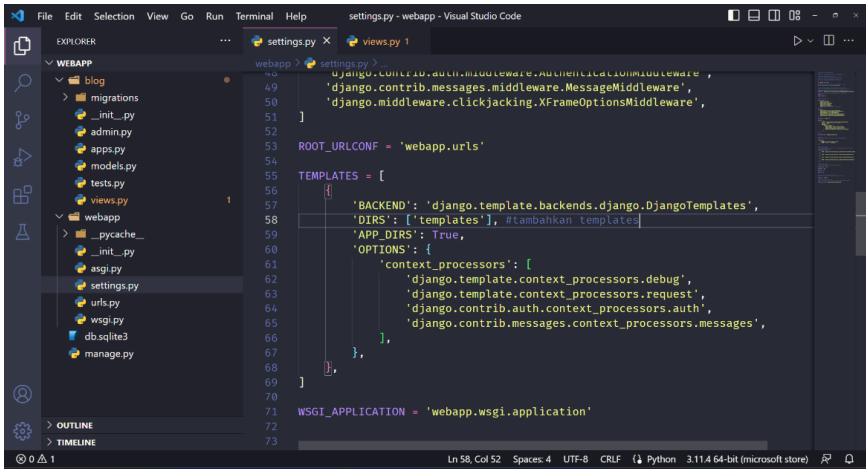
Gambar 50. Membuat Website Hello World

Lalu masuk kedalam folder **blog** dan buka file **views.py**

Lalu buat function baru

```
def index(request):
    context = {
        'title': "Hello World"
    }
    return render(request, 'index.html', context)
```

lalu Kembali ke file setting.py dan tambahkan sebagai berikut:



**Gambar 51. File Setting**

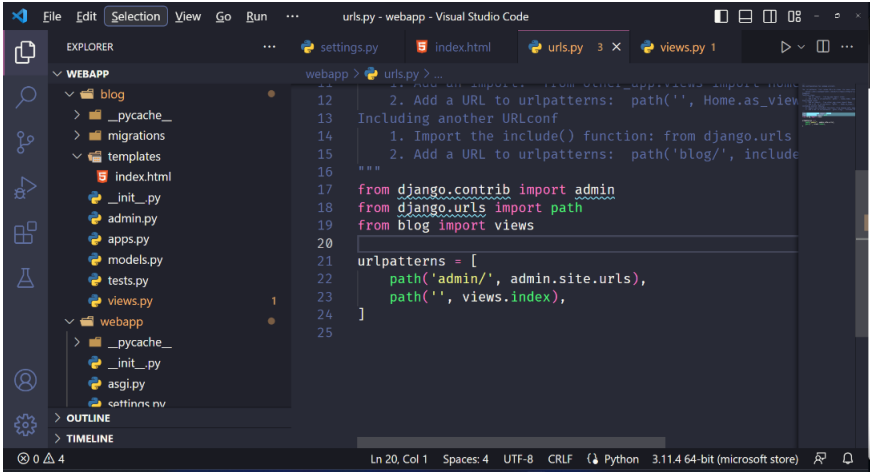
Lalu buat folder templates pada folder **blog/templates** lalu buat juga file **index.html** didalam folder tersebut dan tambahkan syntax dengan Bahasa HTML, sesuai dengan desain yang kamu rancang, sebagai contoh tambahkan :

```

<!DOCTYPE html>
<html lang="en">
<head>
    <title>File Hello
World</title>
</head>
<body>
    <h1>{{ title }}</h1>
</body>
</html>

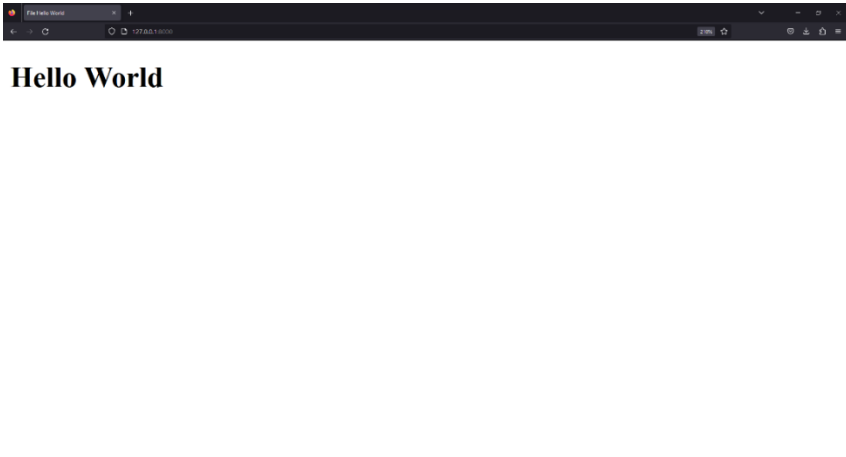
```

Lalu setting routing pada website kita agar mengarah ke index html yang dibuat dengan cara buka file **urls.py** pada folder webapp  
Lalu tambahkan sebagai berikut



**Gambar 52. File urls.py**

Lalu buka browser dengan memasukkan url <http://127.0.0.1:8000/>, hasilnya akan menampilkan output sesuai dengan **index.html** yang sudah dibuat tadi, sebagai berikut :



**Gambar 53. Tampilan Hasil Website Hello World**

## URL dan Views

Dalam framework Django, URL dan view berfungsi untuk mengatur bagaimana permintaan (request) dari pengguna dipetakan ke fungsi (view) yang sesuai untuk menghasilkan



respons (response) yang tepat. URL berperan sebagai peta untuk mengarahkan permintaan ke view yang benar.

- a. URL (Uniform Resource Locator): URL adalah singkatan dari Uniform Resource Locator, yang merupakan alamat yang digunakan untuk mengakses suatu sumber daya di web. Dalam konteks Django, URL menentukan bagaimana permintaan dari pengguna (seperti mengakses halaman web atau mengirim data form) dipetakan ke view yang akan menangani permintaan tersebut.
- b. URL dalam Django didefinisikan dalam berkas `urls.py` yang ada di setiap aplikasi atau di berkas utama proyek Django (biasanya bernama `urls.py` juga). URL berisi pola (pattern) untuk mengenali pola permintaan tertentu dan menentukan view yang akan menangani permintaan tersebut.

Contoh penggunaan URL di berkas `urls.py`:

```
from django.urls import path
from . import views

urlpatterns = [
    path('home/', views.home_view,
        name='home'),
    path('products/',
        views.product_list_view,
        name='product_list'),
    path('products/<int:product_id>/',
        views.product_detail_view,
        name='product_detail'),
]
```

### a) View

View adalah fungsi atau kelas yang menangani permintaan dari pengguna dan mengembalikan respons HTTP. View bertanggung jawab untuk memproses data,

melakukan operasi bisnis, dan memutuskan respons apa yang akan dikirimkan kembali ke pengguna.

View dalam Django dapat ditulis dalam bentuk fungsi atau menggunakan pendekatan berorientasi objek dengan menggunakan kelas berbasis View atau APIView (untuk REST framework).

### **Contoh fungsi view:**

```
from django.shortcuts import render
from django.http import HttpResponse

def home_view(request):
    return HttpResponse("Welcome to the
homepage!")

def product_list_view(request):
    # Logic untuk menampilkan daftar
    produk
    return render(request,
'product_list.html', context)

def product_detail_view(request,
product_id):
    # Logic untuk menampilkan detail
    produk berdasarkan product_id
    return render(request,
'product_detail.html', context)
```

**Contoh view berbasis class :**

```
from rest_framework.views import APIView
from rest_framework.response import
    Response
from rest_framework import status

class ProductListView(APIView):
    def get(self, request):
        # Logic untuk mendapatkan daftar
        produk
        return Response(data,
            status=status.HTTP_200_OK)

    def post(self, request):
        # Logic untuk menambahkan produk
        baru
        return Response (data,
            status=status.HTTP_201_CREATED)
```



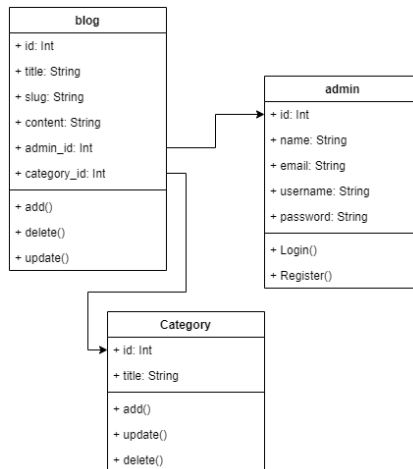
# BAB V

## PROJECT DJANGO MEMBUAT BLOG PRIBADI

### Inisiasi Project

Bab ini akan membahas tentang pembuatan web Blog Pribadi dengan menggunakan django dan database mysql, database mysql merupakan database yang banyak digunakan oleh developer developer saat ini, kita asusmsikan nama database mysql nya adalah django\_blog

Kita akan membuat blog pribadi dengan class diagram dibawah ini

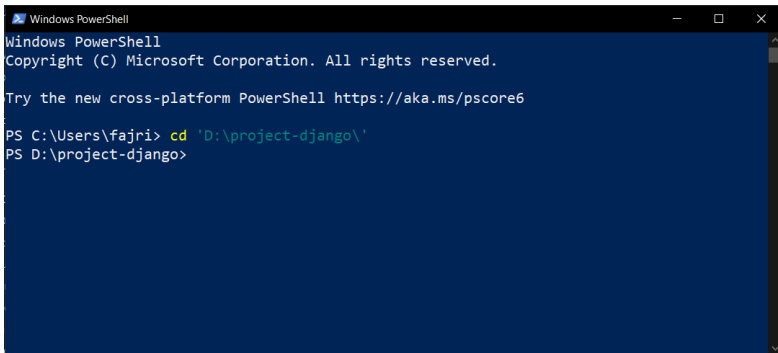


Gambar 54. Class Diagram

## Membuat virtual environment

Langkah-langkah untuk membuat lingkungan virtual (virtual Environment) sebagai berikut

1. Buat folder untuk letak dari project ini, saya asumsikan diletakkan di D:\project-django
2. Masuk ke Powershell di laptop/PC mu
3. Lalu masuk ke folder project yang sudah dibuat tadi



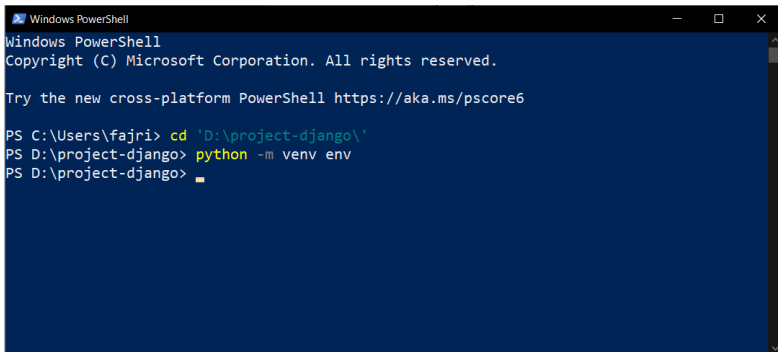
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\fajri> cd 'D:\project-django\'
PS D:\project-django>
```

Gambar 55. Windows Powershell

4. Kemudian ketik perintah `python -m venv env` dan tekan tombol enter



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\fajri> cd 'D:\project-django\'
PS D:\project-django> python -m venv env
PS D:\project-django>
```

Gambar 56. Windows Powershell python -m venv env

5. Akan terlihat folder `env` di dalam folder D:\project-django

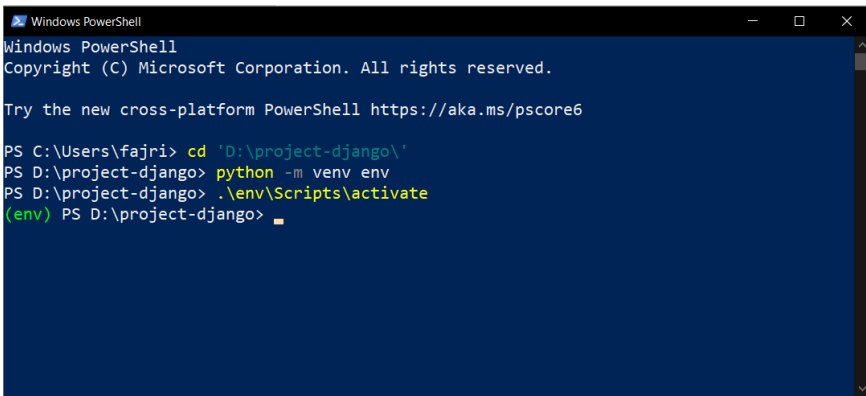
## Mengaktifkan virtual environment

Langkah langkah untuk mengkatifkan atau masuk ke virtual environment sebagai berikut :

1. Untuk mengaktiifkan lingkungan virtual environmnet, pertama harus masuk ke dalam folder tempat env dibuat tadi, lalu ketikkan perintah di powershell sebagai berikut:

`.\env\Scripts\activate`

Sehingga tampilan di powershell akan menjadi seperti ini:

A screenshot of a Windows PowerShell terminal window. The window title is "Windows PowerShell". The text inside shows the following commands and output:

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\fajri> cd 'D:\project-django\'
PS D:\project-django> python -m venv env
PS D:\project-django> .\env\Scripts\activate
(env) PS D:\project-django> _
```

Gambar 57. Windows Activate

## Instalasi dan membuat project Django

Pada project ini kita akan menggunakan python dengan menginstall python versi terbaru saat buku ini ditulis. Langkah-langkah nya sebagai berikut:

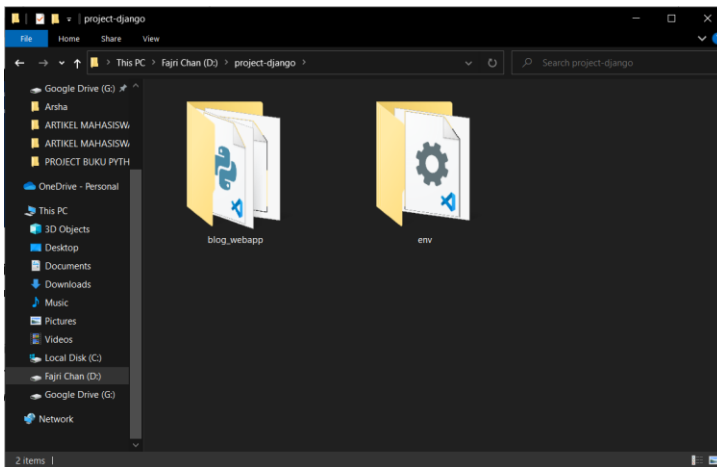
1. Menuju powershell dan ketik perintah dibawah dibawah ini:

*pip install django* kemudian tekan tombol enter  
*sehingga akan terlihat seperti gambar berikut ini:*

```
Windows PowerShell
PS D:\project-django> .\env\Scripts\activate
(env) PS D:\project-django> pip install django
Collecting django
  Using cached Django-4.2.3-py3-none-any.whl (8.0 MB)
Collecting sqlparse>=0.3.1
  Using cached sqlparse-0.4.4-py3-none-any.whl (41 kB)
Collecting asgiref<4,>=3.6.0
  Using cached asgiref-3.7.2-py3-none-any.whl (24 kB)
Collecting tzdata
  Using cached tzdata-2023.3-py2.py3-none-any.whl (341 kB)
Collecting typing-extensions>=4
  Using cached typing_extensions-4.7.1-py3-none-any.whl (33 kB)
Installing collected packages: tzdata, typing-extensions, sqlparse, asgiref, django
Successfully installed asgiref-3.7.2 django-4.2.3 sqlparse-0.4.4 typing-extensions-4.7.1 tzdata-2023.3
WARNING: You are using pip version 22.0.4; however, version 23.2.1 is available.
You should consider upgrading via the 'D:\project-django\env\Scripts\python.exe -m pip install --upgrade pip' command.
(env) PS D:\project-django>
```

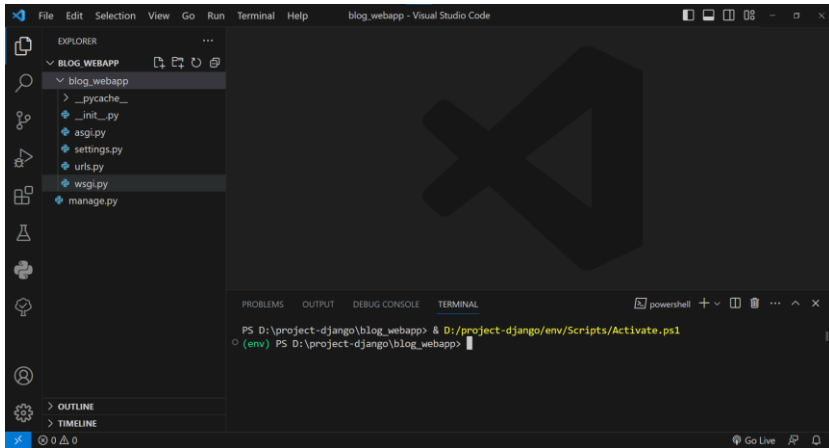
**Gambar 58. Pip Install Django**

2. Lalu untuk membuat project django masukkan perintah berikut:  
*Django-admin startproject blog\_webapp*  
maka akan terlihat folder project django kita pada D:\project-django seperti berikut ini:



**Gambar 59. Folder Django**

3. Lalu masuk buka folder project yang sudah kita buat ke aplikasi visual studio code, dengan perintah  
*Code toko*  
Maka visual studio code aka otomatis terbuka beserta dengan workspace dari folder kita, seperti gambar berikut ini



Gambar 60. Blog WebApp

## Instalasi Library pendukung

Setelah membuat project django kita, kita membutuhkan library pendukung untuk development web blog pribadi yang kita buat ini, sebagai contoh kita memerlukan library *django-ckeditor*, *mysqlclient*, *Pillow*, *django-widget-tweaks*

### Menginstall Library MySQLClient pada project django

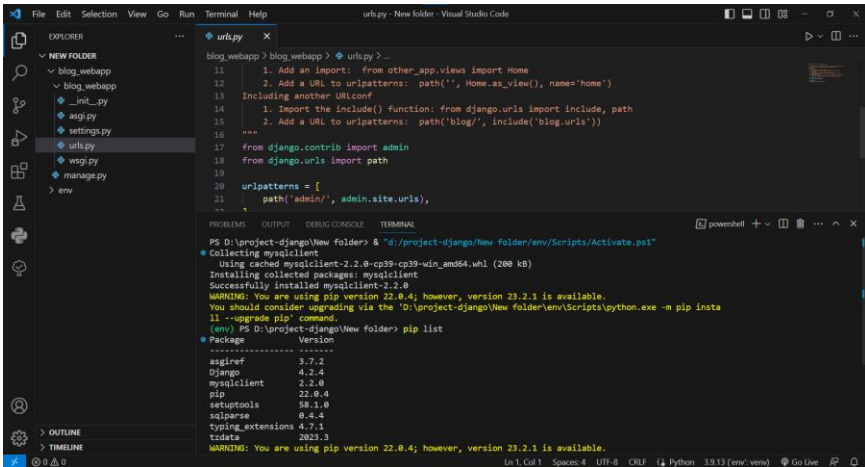
Langkah langkah menginstall paket library MySQLClient sebagai berikut:

1. Buka kembali visual studio code lalu, lalu buka terminal dengan mengklik menu **terminal > New Terminal** atau bisa dengan shortcut tombol **ctrl + shift + `**
2. Ketik perintah

Pip install mysqlclient

lalu tekan tombol enter, dan secara otomatis akan menginstall library mysqlclient pada project django kita, seperti gambar dibawah ini





**Gambar 61. Terminal My Library Sql Client**

## Menginstall Library *django-ckeditor* pada project django

Langkah langkah menginstall paket library Django-ckeditor sebagai berikut:

1. Buka kembali visual studio code lalu, lalu buka terminal dengan mengklik menu **terminal > New Terminal** atau bisa dengan shortcut tombol **ctrl + shift + `**
2. Ketik perintah:

Pip install django-ckeditor

lalu tekan tombol enter, dan secara otomatis akan menginstall library django-ckeditor pada project django kita

## Menginstall Library *Pillow* pada project django

Langkah langkah menginstall paket library Pillow sebagai berikut:

1. Buka kembali visual studio code lalu, lalu buka terminal dengan mengklik menu **terminal > New Terminal** atau bisa dengan shortcut tombol **ctrl + shift + `**
2. Ketik perintah

*Pip install Pillow*

lalu tekan tombol enter, dan secara otomatis akan menginstall library Pillow pada project django kita

## Menginstall Library `django-widget-tweaks` pada project `django`

Langkah langkah menginstall paket library `django-widget-tweaks` sebagai berikut:

1. Buka kembali visual studio code lalu, lalu buka terminal dengan mengklik menu **terminal > New Terminal** atau bisa dengan shortcut tombol **ctrl + shift + `**
2. Ketik perintah

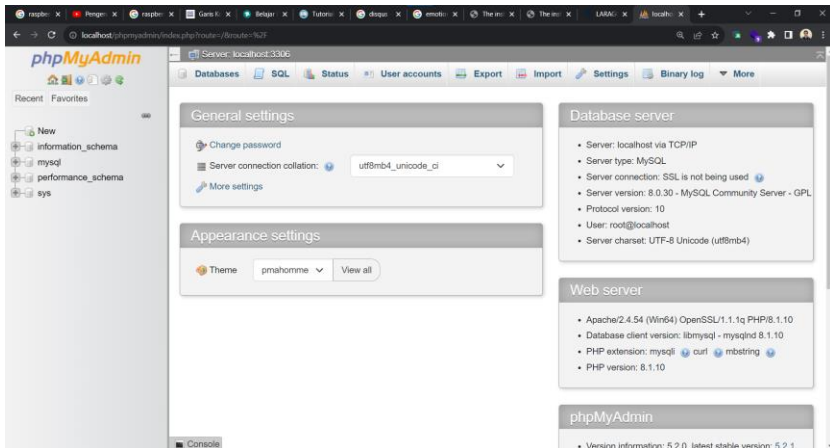
*Pip install django-widget-tweaks*

lalu tekan tombol enter, dan secara otomatis akan menginstall library `django-widget-tweaks` pada project `django` kita

## Membuat database

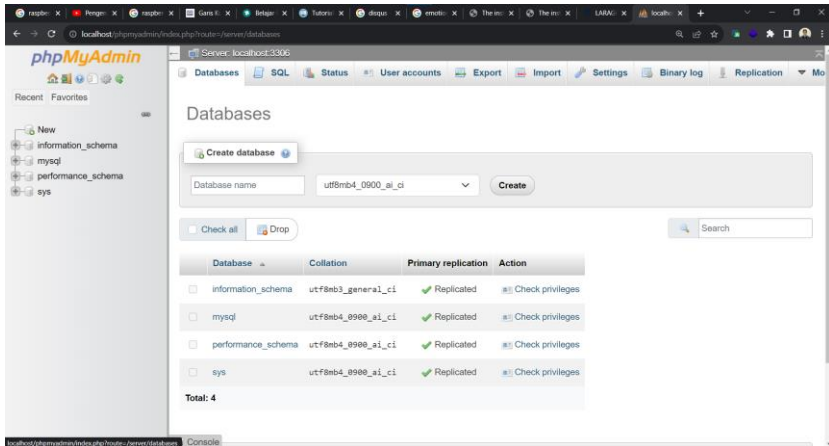
Pada cara pembuatan database ini kita akan menggunakan database `mysql` dengan menamai database kita dengan nama **`toko_db`**, berikut ini adalah langkah langkah dalam pembuatan database sebagai berikut :

1. Buka `xampp` lalu hidupkan `apache` dan `mysql` pada `xampp` anda
  2. Buka `phpmyadmin` pada `Xampp` komputer anda
- Maka akan menampilkan `phpmyadmin` pada browser seperti ini:



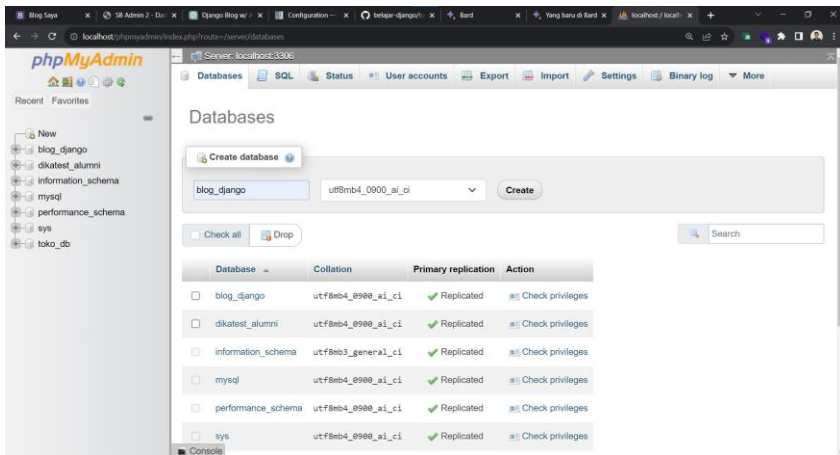
**Gambar 62. Membuat Database**

### 3. Klik tab databases



Gambar 63. Tab Database

### 4. Lalu isikan nama database kita yaitu **blog-django** pada formfield **database name**, lalu tekan **create**



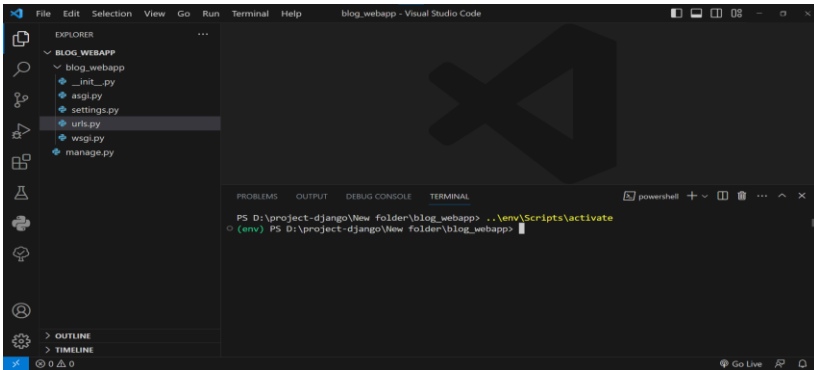
Gambar 64. Mengisikan Nama Database

### 5. Lalu secara otomatis database akan terbuat

## Membuat app blog

Langkah-langkah membuat app toko\_app sebagai berikut

1. Pada Vscode yang sudah dibuka sebelumnya, lalu buka terminal dengan mengklik menu **terminal > New Terminal** atau bisa dengan shortcut tombol **ctrl + shift + `** Sehingga akan muncul tampilan terminal seperti ini

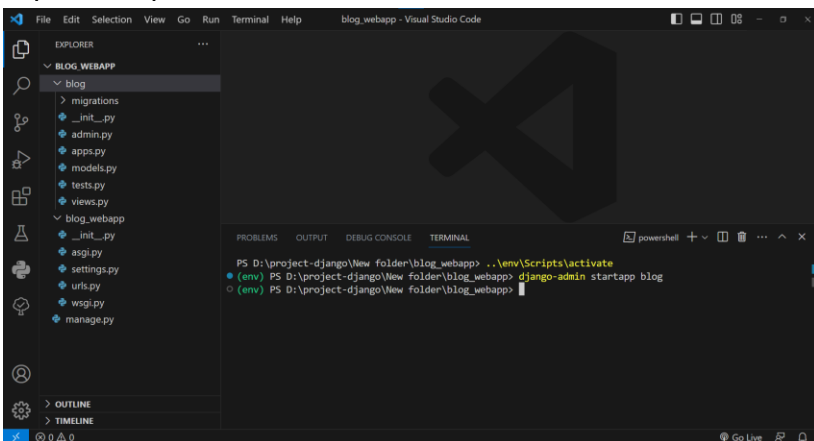


**Gambar 65. Membuat App Blog**

2. Lalu untuk membuat app toko\_app dengan memasukkan perintah sebagai berikut:

Django-admin startapp blog

Lalu akan otomatis terbuat folder app baru di project kita, seperti tampilan berikut ini:

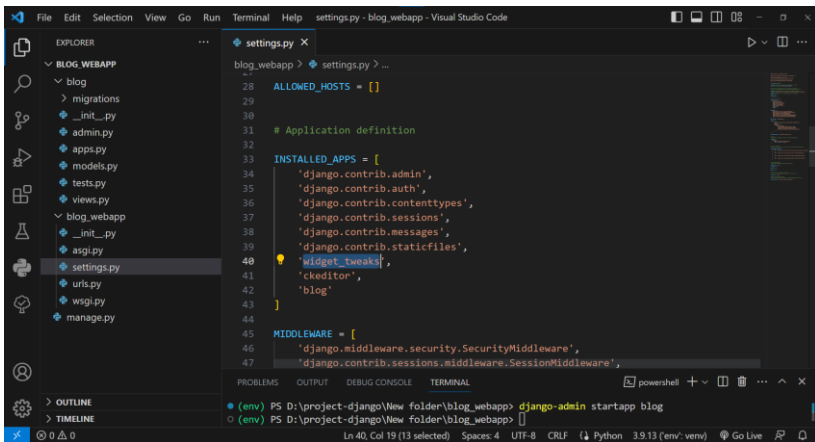


**Gambar 66. Django Admin**

## Mengatur file settings.py pada variable INSTALLED\_APPS dan DATABASE

Langkah langkah mengatur file setting.py sebagai berikut:

1. Klik file setting.py di visual studio code pada project toko
2. Cari variable **INSTALLED\_APPS**
3. Tambahkan variable **'blog'**, **'ckeditor'**, **'widget-tweaks'** kedalam list **INSTALLED\_APPS** seperti berikut

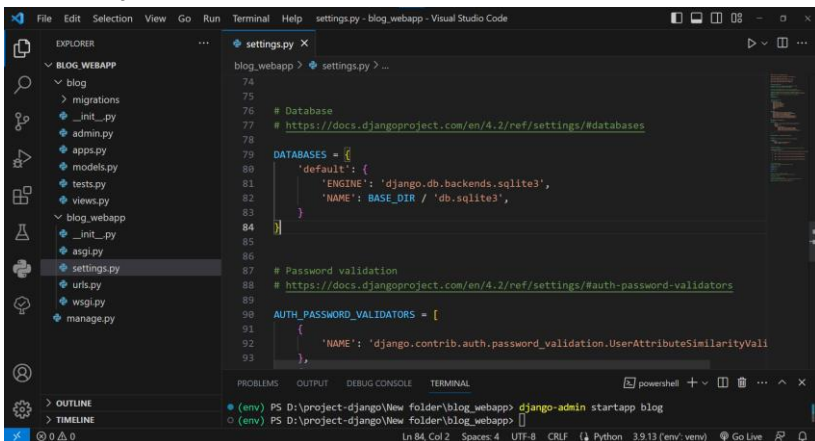


```
blog_webapp > settings.py > ...
28 ALLOWED_HOSTS = []
29
30
31 # Application definition
32
33 INSTALLED_APPS = [
34     'django.contrib.admin',
35     'django.contrib.auth',
36     'django.contrib.contenttypes',
37     'django.contrib.sessions',
38     'django.contrib.messages',
39     'django.contrib.staticfiles',
40     'widget_tweaks',
41     'ckeditor',
42     'blog'
43 ]
44
45 MIDDLEWARE = [
46     'django.middleware.security.SecurityMiddleware',
47     'django.contrib.sessions.middleware.SessionMiddleware',
48 ]
```

Terminal output:  
PS D:\project-django\New folder\blog\_webapp> django-admin startapp blog  
PS D:\project-django\New folder\blog\_webapp>

Gambar 67. Setting.py

4. Jangan lupa disave dengan shortcut tombol **CTRL + S**
5. Lalu cari juga variable **DATABASE** ,seperti ini:



```
74
75
76 # Database
77 # https://docs.djangoproject.com/en/4.2/ref/settings/#databases
78
79 DATABASES = {
80     'default': {
81         'ENGINE': 'django.db.backends.sqlite3',
82         'NAME': BASE_DIR / 'db.sqlite3',
83     }
84 }
85
86 # Password validation
87 # https://docs.djangoproject.com/en/4.2/ref/settings/#auth-password-validators
88
89 AUTH_PASSWORD_VALIDATORS = [
90     {
91         'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityVali
92     },
93 ]
```

Terminal output:  
PS D:\project-django\New folder\blog\_webapp> django-admin startapp blog  
PS D:\project-django\New folder\blog\_webapp>

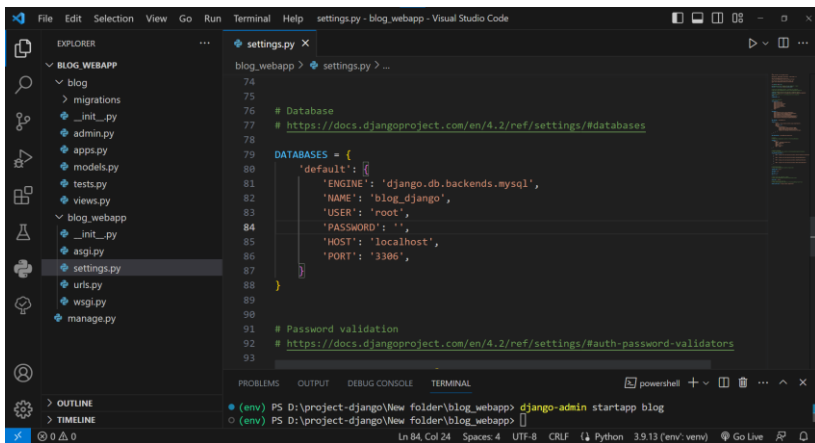
Gambar 68. Variable Database

6. Lalu ubah variabel DATABASE menjadi seperti ini;

```
DATABASES = {
    'default': {
        'ENGINE':
        'django.db.backends.mysql',
        'NAME': 'toko_db',
        'USER': 'root',
        'PASSWORD': ' ',
        'HOST': 'localhost',
        'PORT': '3306',
    }
}
```

Pastikan benar dengan informasi dari database dan mysql di laptop anda, jika misal mysql anda terdapat password silahkan diisi sesuai dengan password mysql anda

Maka akan seperti ini:

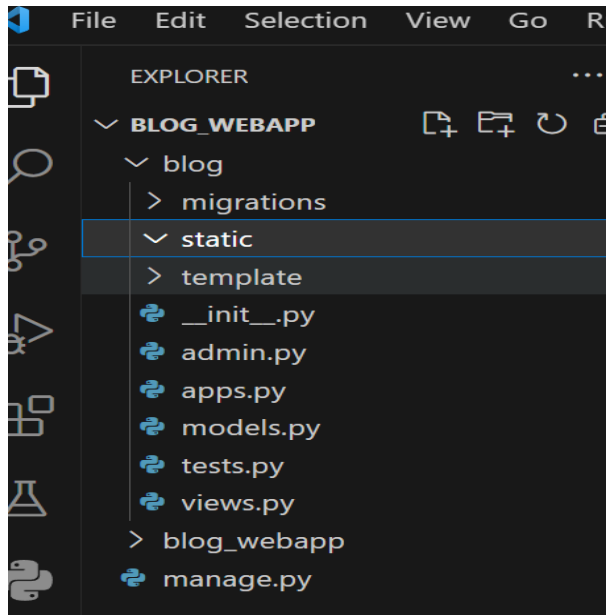


Gambar 69. Password localhost

## Menambahkan folder Template dan folder

Langkah untuk membuat folder template dan folder static sebagai berikut:

1. Pada visual studio code, klik kanan di folder toko dan pilih menu new folder
2. Ketik template dan tekan enter
3. Lalu, lakukan ulang dengan menamai static  
Maka hasilnya akan seperti berikut ini:



**Gambar 70. Menambahkan Folder Template**

4. Lalu tambahkan file-file static ke dalam folder static, silahkan kirim email kepada kami, untuk mendapatkan file static nya  
Email : [fajrinaldichan@gmail.com](mailto:fajrinaldichan@gmail.com)

## Membuat file base.html didalam folder templates/blog

Langkah-langkah membuat file base html di dalam folder templates/blog sebagai berikut:

1. Klik kanan folder blog di bawah folder templates dan pilih menu new file
2. Ketik base.html dan tekan tombol enter

3. Ketik perintah seperti di bawah ini:

```
<!DOCTYPE html>
<html lang="en">

{% load static %}
{{ form.media }}

<head>
  <meta charset="utf-8">
  <meta content="width=device-width, initial-
scale=1.0" name="viewport">

  <title>Blog Saya</title>
  <meta content="" name="description">
  <meta content="" name="keywords">

  <!-- Favicons -->
  <link href="{% static 'blog/img/favicon.png' %}" "
rel="icon">
  <link href="{% static 'blog/img/apple-touch-
icon.png' %}" " rel="apple-touch-icon">

  <!-- Google Fonts -->
  <link
href="https://fonts.googleapis.com/css?family=Open+Sa
ns:300,300i,400,400i,600,600i,700,700i|Jost:300,300i,
400,400i,500,500i,600,600i,700,700i|Poppins:300,300i,
400,400i,500,500i,600,600i,700,700i"
rel="stylesheet">

  <!-- Vendor CSS Files -->
  <link href="{% static 'blog/vendor/aos/aos.css' %}"
" rel="stylesheet">
```



```

    <link href="{% static
'blog/vendor/bootstrap/css/bootstrap.min.css' %}" "
rel="stylesheet">
    <link href="{% static 'blog/vendor/bootstrap-
icons/bootstrap-icons.css' %}" " rel="stylesheet">
    <link href="{% static
'blog/vendor/boxicons/css/boxicons.min.css' %}" "
rel="stylesheet">
    <link href="{% static
'blog/vendor/glightbox/css/glightbox.min.css' %}" "
rel="stylesheet">
    <link href="{% static
'blog/vendor/remixicon/remixicon.css' %}" "
rel="stylesheet">
    <link href="{% static 'blog/vendor/swiper/swiper-
bundle.min.css' %}" " rel="stylesheet">

    <!-- Template Main CSS File -->
    <link href="{% static 'blog/css/style.css' %}" "
rel="stylesheet">
</head>

<body>

{% include 'blog/header.html' %}

{% block body %}{% endblock %}

<!-- ===== Footer ===== -->
<footer id="footer">

    <div class="footer-newsletter">
        <div class="container">
            <div class="row justify-content-center">

```

```

    <div class="col-lg-6">
      <h4>Join Our Newsletter</h4>
      <p>Lorem ipsum dolor sit amet consectetur
adipiscing elit. Velit eveniet odio consequuntur
earum dicta impedit.</p>
      <form action="" method="post">
        <input type="email" name="email"><input
type="submit" value="Subscribe">
      </form>
    </div>
  </div>
</div>

```

```

<div class="footer-top">
  <div class="container">
    <div class="row">

      <div class="col-lg-3 col-md-6 footer-
contact">
        <h3>Blog Saya</h3>
        <p>
          Jl. Gurun Aua, Kubang Putih, Kec.
Banuhampu, Kota Bukittinggi, Sumatera Barat 26181
<br><br>
          <strong>Phone:</strong> +62896-0000-
0000<br>
          <strong>Email:</strong>
admin@orbituinbkt.com<br>
        </p>
      </div>

```

```

    <div class="col-lg-3 col-md-6 footer-
links">

```

```

        <h4>Useful Links</h4>
        <ul>
            <li><i class="bx bx-chevron-right"></i>
<a href="#">Home</a></li>
            <li><i class="bx bx-chevron-right"></i>
<a href="#">About us</a></li>
            <li><i class="bx bx-chevron-right"></i>
<a href="#">Services</a></li>
            <li><i class="bx bx-chevron-right"></i>
<a href="#">Terms of service</a></li>
            <li><i class="bx bx-chevron-right"></i>
<a href="#">Privacy policy</a></li>
        </ul>
    </div>

```

```

<div class="col-lg-3 col-md-6 footer-
links">

```

```

        <h4>Our Services</h4>
        <ul>
            <li><i class="bx bx-chevron-right"></i>
<a href="#">Web Design</a></li>
            <li><i class="bx bx-chevron-right"></i>
<a href="#">Web Development</a></li>
            <li><i class="bx bx-chevron-right"></i>
<a href="#">Product Management</a></li>
            <li><i class="bx bx-chevron-right"></i>
<a href="#">Marketing</a></li>
            <li><i class="bx bx-chevron-right"></i>
<a href="#">Graphic Design</a></li>
        </ul>
    </div>

```

```

<div class="col-lg-3 col-md-6 footer-
links">

```

```

        <h4>Our Social Networks</h4>
        <p>Cras fermentum odio eu feugiat lide
par naso tierra videa magna derita valies</p>
        <div class="social-links mt-3">
            <a href="#" class="twitter"><i
class="bx bxl-twitter"></i></a>
            <a href="#" class="facebook"><i
class="bx bxl-facebook"></i></a>
            <a href="#" class="instagram"><i
class="bx bxl-instagram"></i></a>
            <a href="#" class="google-plus"><i
class="bx bxl-skype"></i></a>
            <a href="#" class="linkedin"><i
class="bx bxl-linkedin"></i></a>
        </div>
    </div>
</div>
</div>

<div class="container footer-bottom clearfix">
    <div class="copyright">
        &copy; Copyright. All Rights Reserved
    </div>
</div>
</footer><!-- End Footer -->

<!-- Vendor JS Files -->
<script src="{% static 'blog/vendor/aos/aos.js' %}
"></script>

```

```

    <script src="{% static
'blog/vendor/bootstrap/js/bootstrap.bundle.min.js' %}
"></script>
    <script src="{% static
'blog/vendor/glightbox/js/glightbox.min.js' %}
"></script>
    <script src="{% static 'blog/vendor/isotope-
layout/isotope.pkgd.min.js' %} "></script>
    <script src="{% static 'blog/vendor/swiper/swiper-
bundle.min.js' %} "></script>
    <script src="{% static
'blog/vendor/waypoints/noframework.waypoints.js' %}
"></script>
    <script src="{% static 'blog/vendor/php-email-
form/validate.js' %} "></script>

    <!-- Template Main JS File -->
    <script src="{% static 'blog/js/main.js' %}
"></script>

</body>
</html>

```

## Membuat file header.html di dalam folder templates/blog

Langkah-langkah membuat file header di dalam folder templates/blog sebagai berikut:

1. Klik kanan folder blog di bawah folder templates dan pilih menu new file
2. Ketik header.html dan tekan tombol enter
3. Ketik perintah seperti di bawah ini:

```

<header id="header" class="fixed-top {% if
request.path != '/' %} header-inner-pages {% endif %}
">

```

```

<div class="container d-flex align-items-center">

    <h1 class="logo me-auto"><a href="{% url 'home'
%}">BLOG SAYA</a></h1>
    <!-- Uncomment below if you prefer to use an
image logo -->
    <!-- <a href="index.html" class="logo me-
auto"></a>-->

    <nav id="navbar" class="navbar">
        <ul>
            <li><a class="nav-link scrollto" href="{%
url 'home' %}">Home</a></li>
            <li><a class="nav-link scrollto"
href="#about">About Me</a></li>
            <li><a class="nav-link scrollto"
href="#services">Services</a></li>
            <li class="dropdown "><a href="{% url
'blog' %}"><span>Blog</span> <i class="bi bi-chevron-
down"></i></a>
                <ul>
                    <li><a href="{% url 'blog'
%}">ALL</a></li>

                    {% for category in categories %}
                    <li><a href="{% url
'category_blog_list' category.id
%}">{{category}}</a></li>
                    {% endfor %}

                </ul>
            </li>
        </ul>
    </nav>

```

```

        <li><a class="nav-link scrollto"
href="#contact">Contact</a></li>
    </ul>
    <i class="bi bi-list mobile-nav-toggle"></i>
</nav><!-- .navbar -->

</div>
</header>

```

## Membuat file home.html di dalam folder templates/blog

Langkah-langkah membuat file header di dalam folder templates/blog sebagai berikut:

1. Klik kanan folder blog di bawah folder templates dan pilih menu new file
2. Ketik home.html dan tekan tombol enter
3. Ketik perintah seperti di bawah ini:

```

{% extends 'blog/base.html' %}
{% block body %}
{% load static %}

```

```

<!-- ===== Hero Section ===== -->
<section id="hero" class="d-flex align-items-center">

    <div class="container">
        <div class="row">
            <div class="col-lg-6 d-flex flex-column justify-content-center pt-4 pt-lg-0 order-2 order-lg-1" data-aos="fade-up" data-aos-delay="200">
                <h1>Selamat datang di</h1>
                <h2>Blog Saya</h2>
                <div class="d-flex justify-content-center justify-content-lg-start">

```

```
        <a href="#about" class="btn-get-started
scrollto">Get Started</a>
    </div>
</div>
    <div class="col-lg-6 order-1 order-lg-2 hero-
img" data-aos="zoom-in" data-aos-delay="200">
        
    </div>
</div>
</div>
```

```
</section><!-- End Hero -->
```

```
<main id="main">
```

```
    <!-- ===== About Section ===== -->
    <section id="about" class="skills">
        <div class="container card p-3" data-aos="fade-
up">
```

```
            <div class="row">
                <div class="col-lg-6 d-flex align-items-
center" data-aos="fade-right" data-aos-delay="100">
                    
                </div>
```

```
                <div class="col-lg-6 pt-4 pt-lg-0 content"
data-aos="fade-left" data-aos-delay="100">
                    <h3>Tentang Saya</h3>
                    <p class="fst-italic">
                        Lorem ipsum dolor sit, amet consectetur
adipisicing elit. Magnam officiiis neque aspernatur
voluptatibus culpa tenetur nam repellat nostrum
```



recusandae debitis inventore, esse sint laboriosam voluptatum dolorum pariatu fugiat est exercitationem itaque reprehenderit iste labore nihil ea dolores! Debitis molestias ut a aut omnis dolore, ullam sit! Atque, quam! Alias nihil molestiae maxime ratione, cumque possimus nemo dolorum totam laborum quibusdam similique facere debitis mollitia assumenda consecetur sed reprehenderit delectus voluptas.

```
        </p>
        <a href="#" class="btn-learn-more">Learn
More</a>
    </div>
</div>
```

```
</div>
</section><!-- End About Section -->
```

```
<!-- ===== Services Section ===== -->
<section id="services" class="services section-
bg">
```

```
    <div class="container" data-aos="fade-up">

        <div class="section-title">
            <h2>Services</h2>
            <p>Magnam dolores commodi suscipit.
Necessitatibus eius consequatur ex aliquid fuga eum
quidem. Sit sint consecetur velit. Quisquam quos
quisquam cupiditate. Et nemo qui impedit suscipit
alias ea. Quia fugiat sit in iste officiis commodi
quidem hic quas.</p>
        </div>
```

```
    <div class="row">
```

```

        <div class="col-xl-3 col-md-6 d-flex
align-items-stretch" data-aos="zoom-in" data-aos-
delay="100">
            <div class="icon-box">
                <div class="icon"><i class="bx bxl-
dribbble"></i></div>
                <h4><a href="">Lorem Ipsum</a></h4>
                <p>Voluptatum deleniti atque corrupti
quos dolores et quas molestias excepturi</p>
            </div>
        </div>

```

```

        <div class="col-xl-3 col-md-6 d-flex
align-items-stretch mt-4 mt-md-0" data-aos="zoom-in"
data-aos-delay="200">
            <div class="icon-box">
                <div class="icon"><i class="bx bx-
file"></i></div>
                <h4><a href="">Sed ut
perspici</a></h4>
                <p>Duis aute irure dolor in
reprehenderit in voluptate velit esse cillum
dolore</p>
            </div>
        </div>

```

```

        <div class="col-xl-3 col-md-6 d-flex
align-items-stretch mt-4 mt-xl-0" data-aos="zoom-in"
data-aos-delay="300">
            <div class="icon-box">
                <div class="icon"><i class="bx bx-
tachometer"></i></div>
                <h4><a href="">Magni Dolores</a></h4>

```

```
        <p>Excepteur sint occaecat cupidatat  
non proident, sunt in culpa qui officia</p>  
    </div>  
</div>
```

```
    <div class="col-xl-3 col-md-6 d-flex  
align-items-stretch mt-4 mt-xl-0" data-aos="zoom-in"  
data-aos-delay="400">  
        <div class="icon-box">  
            <div class="icon"><i class="bx bx-  
layer"></i></div>  
            <h4><a href="">Nemo Enim</a></h4>  
            <p>At vero eos et accusamus et iusto  
odio dignissimos ducimus qui blanditiis</p>  
        </div>  
    </div>
```

```
</div>
```

```
</div>
```

```
</section><!-- End Services Section -->
```

```
<!-- ===== Cta Section ===== -->
```

```
<section id="cta" class="cta">
```

```
    <div class="container" data-aos="zoom-in">
```

```
        <div class="row">
```

```
            <div class="col-lg-9 text-center text-lg-  
start">
```

```
                <h3>Call To Action</h3>
```

```
                <p> Duis aute irure dolor in  
reprehenderit in voluptate velit esse cillum dolore  
eu fugiat nulla pariatur. Excepteur sint occaecat
```

```

cupidatat non proident, sunt in culpa qui officia
deserunt mollit anim id est laborum.</p>
    </div>
    <div class="col-lg-3 cta-btn-container
text-center">
        <a class="cta-btn align-middle"
href="#">Call To Action</a>
    </div>
</div>
</section><!-- End Cta Section -->

<!-- ===== Blog Section ===== -->
<section id="Blog" class="Blog">
    <div class="container" data-aos="fade-up">

        <div class="section-title">
            <h2>Blog</h2>
            <p>Magnam dolores commodi suscipit.
Necessitatibus eius consequatur ex aliquid fuga eum
quidem. Sit sint consectetur velit. Quisquam quos
quisquam cupiditate. Et nemo qui impedit suscipit
alias ea. Quia fugiat sit in iste officiis commodi
quidem hic quas.</p>
        </div>
        <div class="row">
            {% for blog in blogs|slice:":6" %}
            <div class="col-md-4 mb-3">
                <div class="card">
                    
                    <div class="card-body">

```

```

        <h5 class="card-
title">{{blog.title}}</h5>
        <p class="card-
text">{{blog.content|safe|slice:"":100}}...</p>
        <a href="{% url 'blog_detail'
blog.slug %}" class="btn btn-primary">Go
somewhere</a>
    </div>
</div>
</div>
{% endfor %}
</div>

```

```

</div>
</section><!-- End Blog Section -->

```

```

<!-- ===== Contact Section ===== -->
<section id="contact" class="contact">
    <div class="container" data-aos="fade-up">

        <div class="section-title">
            <h2>Contact</h2>
            <p>Magnam dolores commodi suscipit.
Necessitatibus eius consequatur ex aliquid fuga eum
quidem. Sit sint consectetur velit. Quisquam quos
quisquam cupiditate. Et nemo qui impedit suscipit
alias ea. Quia fugiat sit in iste officiis commodi
quidem hic quas.</p>
        </div>

```

```

<div class="row">

  <div class="col-lg-5 d-flex align-items-
stretch">

    <div class="info">
      <div class="address">
        <i class="bi bi-geo-alt"></i>
        <h4>Location:</h4>
        <p>Jl. Gurun Aua, Kubang Putiah,
Kec. Banuhampu, Kota Bukittinggi, Sumatera Barat
26181</p>
      </div>

      <div class="email">
        <i class="bi bi-envelope"></i>
        <h4>Email:</h4>
        <p>admin@orbituinbkt.com</p>
      </div>

      <div class="phone">
        <i class="bi bi-phone"></i>
        <h4>Call:</h4>
        <p>+62896-0000-0000</p>
      </div>

      <iframe
src="https://www.google.com/maps/embed?pb=!1m14!1m8!1
m3!1d12097.433213460943!2d-
74.0062269!3d40.7101282!3m2!1i1024!2i768!4f13.1!3m3!1
m2!1s0x0%3A0xb89d1fe6bc499443!2sDowntown+Conference+C
enter!5e0!3m2!1smk!2sbg!4v1539943755621"
frameborder="0" style="border:0; width: 100%; height:
290px;" allowfullscreen></iframe>
    </div>

```

```

        </div>
        {% load widget_tweaks %}
        <div class="col-lg-7 mt-5 mt-lg-0 d-flex
align-items-stretch">
            {% if messages %}
                {% for message in messages %}
                    <div class="alert alert-success"
role="alert">
                        {{ message }}
                    </div>
                {% endfor %}
            {% endif %}
            <form action="{% url 'inbox_post' %}"
method="get" class="php-email-form">
                {% csrf_token %}
                <div class="row">
                    <div class="form-group col-md-6">
                        <label for="name">Your
Name</label>
                        {{ form.name|add_class:'form-
control' }}
                    </div>
                    <div class="form-group col-md-6">
                        <label for="name">Your
Email</label>
                        {{ form.email|add_class:'form-
control' }}
                    </div>
                </div>
                <div class="form-group">
                    <label for="name">Subject</label>
                    {{ form.subject|add_class:'form-
control' }}

```

```

        </div>
        <div class="form-group">
            <label for="name">Message</label>
            {{ form.message|add_class:'form-
control' }}
        </div>

        <div class="text-center"><button
type="submit">Send Message</button></div>
        </form>
        </div>

    </div>

</div>
</section><!-- End Contact Section -->

</main><!-- End #main -->

{% endblock body %}

```

## Membuat file `blog_detail.html` di dalam folder `templates/blog`

Langkah-langkah membuat file `blog_detail` di dalam folder `templates/blog` sebagai berikut:

1. Klik kanan folder `blog` di bawah folder `templates` dan pilih menu `new file`
2. Ketik `blog_detail.html` dan tekan tombol `enter`
3. Ketik perintah seperti di bawah ini:

```
{% extends 'blog/base.html' %}
```

```
{% block body %}
```



```

{% load static %}
<main id="main">

    <section class="inner-page">
        <div class="container mt-5">
            <header class="blog-header text-center">
                <h1 class="blog-
title">{{blog.title}}</h1>
                <p class="blog-meta">Published on
{{blog.created_at}} by {{blog.admin}}</p>
            </header>

            <article class="blog-content">
                <div class="row">
                    <div class="col-lg-8 offset-lg-2">
                        
                        {{blog.content|safe}}
                    </div>
                </div>
            </article>
        </div>
    </section>
</main>
<!-- End #main -->
{% endblock %}

```

## Mengatur STATIC dan MEDIA ROOT Pada setting.py

Langkah- langkah mengatur file seting.py sebagai berikut:

1. Klik file setting.py di visual studio code pada project blog\_webapp
2. Cari variable STATIC\_URL
3. Modifikasi Seperti gambar berikut:

```

STATIC_URL = 'static/'
MEDIA_URL = '/media/'
MEDIA_ROOT = os.path.join(BASE_DIR,
'media/')

```

## Memodifikasi file urls.py

Langkah- langkah mengatur file seting.py sebagai berikut:

1. Klik file urls.py di visual studio code pada project blog\_webapp
2. Modifikasi Seperti gambar berikut:

```

from django.contrib import admin
from django.urls import path, include
from django.conf import settings
from django.conf.urls.static import static

```

```

urlpatterns = [
    path('admin/', admin.site.urls),
    path("", include('blog.urls')),
    path("adminblog/",
include('admin_blog.urls'))
]

```

```

if settings.DEBUG:
    urlpatterns +=
static(settings.MEDIA_URL,document_root=setting
s.MEDIA_ROOT)

```

3. Lalu pergi ke folder blog lalu klik new file pada di dalam folder blog lalu buat dengan nama urls.py
4. Lalu tambahkan kode seperti berikut ini

```

from django.urls import path
from . import views

urlpatterns = [
    path("", views.home, name="home"),
    path("inboxpost", views.inboxPost,
name="inbox_post"),
    path("blog", views.blog_list,
name="blog"),
    path("blog/<slug:slug>",
views.blog_detail, name="blog_detail"),
    path("categories/<int:id>",
views.category_blog_list,
name="category_blog_list")
]

```

## Memodifikasi file models.py di dalam folder blog

Langkah Langkah memodifikasi file models.py di dalam folder blog sebagai berikut:

1. Klik file models.py di dalam folder blog
2. Ubah kode menjadi seperti berikut

```

from django.db import models
from django.utils.text import slugify
from django.urls import reverse
from ckeditor.fields import RichTextField

# Create your models here.
class category(models.Model):

    id = models.AutoField(primary_key=True)
    title = models.CharField(max_length=50)

```

```

        created_at =
models.DateTimeField(auto_now_add=True)
        updated_at =
models.DateTimeField(auto_now=True)

class Meta:
    verbose_name = ("category")
    verbose_name_plural = ("categories")
    db_table = 'category'

def __str__(self):
    return self.title

def get_absolute_url(self):
    return reverse("category_detail",
kwargs={"pk": self.pk})

class blog(models.Model):

    id = models.AutoField(primary_key=True)
    title = models.CharField( max_length=200)
    thumbnail =
models.ImageField(upload_to='thumbnail/',
blank=True, null=True)
    slug = models.SlugField(max_length=200,
blank=True)
    content = RichTextField()
    admin = models.ForeignKey("admin_blog.admin",
verbose_name=(""), on_delete=models.CASCADE)
    category = models.ForeignKey("category",
verbose_name=(""), on_delete=models.CASCADE)
    created_at =
models.DateTimeField(auto_now_add=True)

```

```

    updated_at =
models.DateTimeField(auto_now=True)
    def save(self, *args, **kwargs):
    if not self.slug:
        self.slug = slugify(self.title)
    super(blog, self).save(*args, **kwargs)

class Meta:
    verbose_name = 'blog'
    verbose_name_plural = 'blogs'
    db_table = 'blog'

def __str__(self):
    return self.title

class inbox(models.Model):

    name = models.CharField(max_length=100)
    email =models.EmailField(max_length=254)
    subject = models.CharField(max_length=254,
blank=True)
    message = models.TextField()

class Meta:
    verbose_name = 'inbox'
    verbose_name_plural = "inboxes"
    db_table = 'inbox'

def __str__(self):
    return self.name

def get_absolute_url(self):

```

```
        return reverse("inbox_detail",
kwargs={"pk": self.pk})
```

## Menambahkan file forms.py di dalam folder blog

Langkah langkah menambah file forms.py di dalam folder blog, sebagai berikut:

1. Klik kanan di folder blog dan pilih new file
2. Ketik forms.py dan tekan tombol enter
3. Masukkan kode sebagai berikut

```
from django import forms
from .models import inbox

class inboxForm(forms.ModelForm):
    class Meta:
        model = inbox
        fields = ['name', 'email', 'subject',
'message']
```

## Memodifikasi file views.py di folder blog

Langkah Langkah memodifikasi file views.py di dalam folder blog sebagai berikut:

1. Klik file models.py di dalam folder blog
2. Ubah kode menjadi seperti berikut:

```
from django.shortcuts import render,
get_object_or_404, redirect
from django.contrib import messages
from .models import blog,category,inbox
from .forms import inboxForm
from django.shortcuts import render,
get_object_or_404, redirect
from django.contrib import messages
from .models import blog,category,inbox
```

```

from .forms import inboxForm

def home(request):
    context = {
        "blogs" : blog.objects.all(),
        "categories" : category.objects.all(),
        "form" : inboxForm()
    }
    print(context)
    return render(request, "blog/home.html",
context)

def inboxPost(request):
    if request.method == 'POST':
        try:
            name = request.POST.get("name")
            email = request.POST.get("email")
            subject = request.POST.get("subject")
            message = request.POST.get("message")
            print(name)
            inbox(name=name, email=email,
subject=subject, message=message)
            messages.success(request, 'Pesan
Berhasil Terkirim!')
            return redirect("home")
        except :
            return redirect("blog")
    else:
        return redirect("home")

def blog_list(request):
    context = {
        "categories" : category.objects.all(),
        "blogs" : blog.objects.all()
    }

```

```

    print(context)
    return render(request, "blog/blog.html",
context)

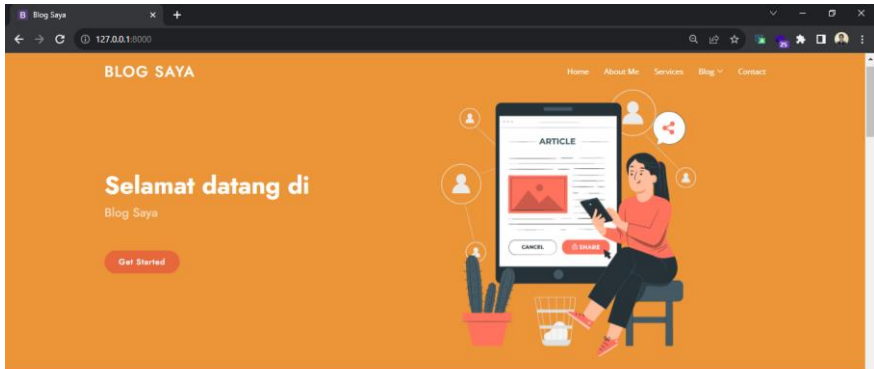
def category_blog_list(request, id):
    context = {
        "name" : category.objects.filter(pk=id),
        "categories" : category.objects.all(),
        "blogs" :
blog.objects.filter(category_id=id)
    }
    print(context)
    return render(request, "blog/blog.html",
context)

def blog_detail(request, slug):
    blog_instance = get_object_or_404(blog,
slug=slug)
    context = {
        "categories" : category.objects.all(),
        "blog": blog_instance
    }
    print(context)
    return render(request,
"blog/blog_detail.html", context)

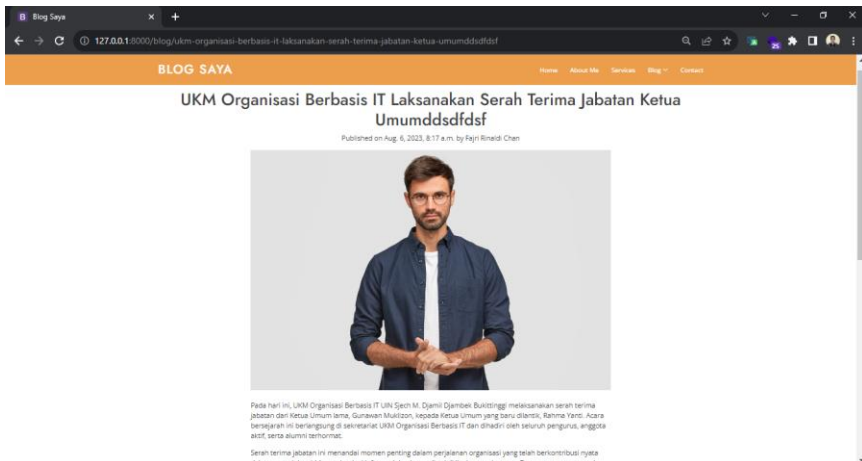
```



Maka hasil dari desain template yang dibuat adalah seperti ini:



**Gambar 71. Halaman Blog Saya**



**Gambar 72. Halaman Content Blog**

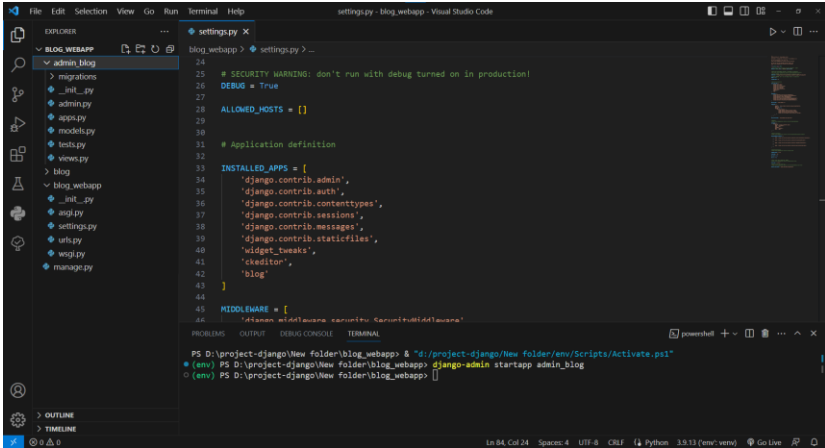
## Membuat app Admin Blog

Langkah membuat app admin\_blog sebagai berikut:

1. Dbuka terminal di visual studio code
2. Lalu ketikkan perintah:

*django-admin startapp admin\_blog*

Maka akan nampak seperti ini

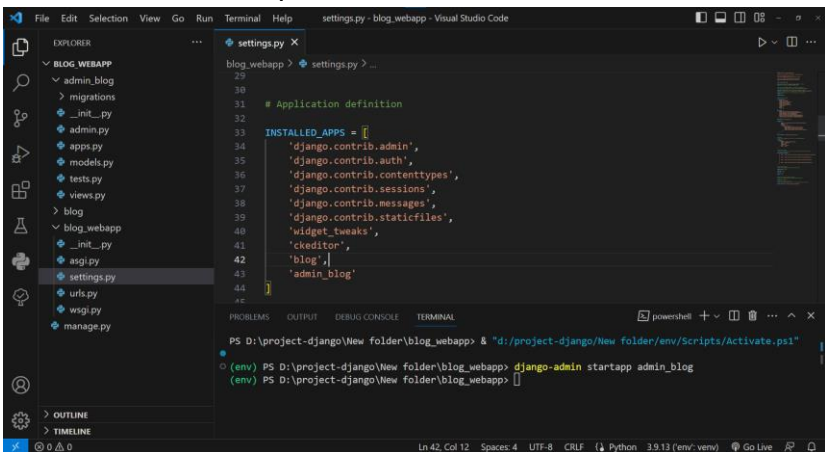


Gambar 73. App Admin Blog

## Menambahkan ‘admion\_blog’ difile settings.py pada variable INSTALLED\_APPS

Lankah langkah mengatur file setting.py sebagai berikut:

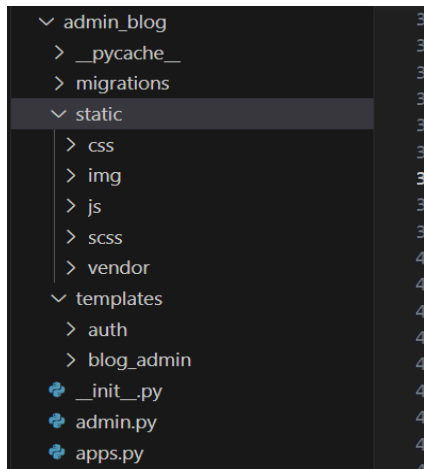
1. Klik file setting.py di visual studio code pada project toko
2. Cari variable **INSTALLED\_APPS**
3. Tambahkan variable ‘**admin\_blog**’ kedalam list **INSTALLED\_APPS** seperti berikut



Gambar 74. Django-Admin

## Membahkan file folder templates dan static di app admin blog

Langkah dalam menabahkan file folder template dan static di app admin blog, sama seperti yang kita lakukan di app blog sebelumnya, hasilnya akan seperti berikut:



**Gambar 75. File Folder Template**

Untuk file static di dalamnya, bisa anda hubungi kami dari email: [fajririnaldichan@gmail.com](mailto:fajririnaldichan@gmail.com)

## Membuat file base.html di dalam folder templates/auth di app admin\_blog

Langkah-langkah membuat file base.html di dalam folder templates/auth di app admin\_blog sebagai berikut:

1. Klik kanan folder auth di bawah folder templates dan pilih menu new file
2. Ketik base.html dan tekan tombol enter
3. Ketik perintah seperti di bawah ini:  

```
<!DOCTYPE html>  
<html lang="en">  
{% load static %}
```

```

<head>

    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible"
content="IE=edge">
    <meta name="viewport" content="width=device-
width, initial-scale=1, shrink-to-fit=no">
    <meta name="description" content="">
    <meta name="author" content="">

    <title>SB Admin 2 - Login</title>

    <!-- Custom fonts for this template-->
    <link href="{% static 'vendor/fontawesome-
free/css/all.min.css' %}" rel="stylesheet"
type="text/css">
    <link

href="https://fonts.googleapis.com/css?family=Nuni
to:200,200i,300,300i,400,400i,600,600i,700,700i,80
0,800i,900,900i"
    rel="stylesheet">

    <!-- Custom styles for this template-->
    <link href="{% static 'css/sb-admin-2.min.css'
%}" rel="stylesheet">

</head>

<body>
    {% block body %}{% endblock %}
    <!-- Bootstrap core JavaScript-->
    <script src="{% static
'vendor/jquery/jquery.min.js' %}" "></script>

```

```

    <script src="{% static
'vendor/bootstrap/js/bootstrap.bundle.min.js'
%}"></script>

    <!-- Core plugin JavaScript-->
    <script src="{% static 'vendor/jquery-
easing/jquery.easing.min.js' %}"></script>

    <!-- Custom scripts for all pages-->
    <script src="{% static 'js/sb-admin-2.min.js'
%}"></script>

</body>
</html>

```

## Membuat file login.html di dalam folder templates/auth di app admin\_blog

Langkah-langkah membuat file login.html di dalam folder templates/auth di app admin\_blog sebagai berikut:

1. Klik kanan folder auth di bawah folder templates dan pilih menu new file
2. Ketik login.html dan tekan tombol enter
3. Ketik perintah seperti di bawah ini:

```

{% extends 'auth/base.html' %}
{% block title %}
    Login
{% endblock %}

{% block body %}
    <div class="container">
        <!-- Outer Row -->
        <div class="row justify-content-center">
            <div class="col-xl-10 col-lg-12 col-md-9">

```

```

<div class="card o-hidden border-0 shadow-
lg my-5">
  <div class="card-body p-0">
    <!-- Nested Row within Card Body -->
    <div class="row">
      <div class="col-lg-6 d-none d-lg-
block bg-login-image"></div>
      <div class="col-lg-6">
        <div class="p-5">
          <div class="text-center">
            <h1 class="h4 text-gray-900
mb-4">Welcome Back!</h1>
          </div>
          {% if messages %}
            {% for message in messages
%}
              <div class="alert
alert-danger" role="alert">
                {{ message }}
              </div>
            {% endfor %}
          {% endif %}
          <form class="user" method="post"
action="{% url 'login_admin' %}">
            {% csrf_token %}
            <div class="form-group">
              <input name="email"
type="email" class="form-control form-control-
user" id="exampleInputEmail" aria-
describedby="emailHelp" placeholder="Enter Email
Address..." />
            </div>
            <div class="form-group">

```

```

        <input name="password"
type="password" class="form-control form-control-
user" id="exampleInputPassword"
placeholder="Password" />
    </div>
    <div class="form-group">
        <div class="custom-control
custom-checkbox small">
            <input type="checkbox"
class="custom-control-input" id="customCheck" />
            <label class="custom-
control-label" for="customCheck">Remember
Me</label>
        </div>
    </div>
    <input type="submit"
class="btn btn-primary btn-user btn-block"
value="Login">
    <hr />

</form>
<hr />
<div class="text-center">
    <a class="small" href="forgot-
password.html">Forgot Password?</a>
</div>
<div class="text-center">
    <a class="small"
href="register.html">Create an Account!</a>
</div>
</div>
</div>
</div>
</div>

```

```

        </div>
    </div>
</div>
</div>
{% endblock %}

```

## Membuat file signup.html di dalam folder templates/auth di app admin\_blog

Langkah-langkah membuat file signup.html di dalam folder templates/auth di app admin\_blog sebagai berikut:

1. Klik kanan folder auth di bawah folder templates dan pilih menu new file
2. Ketik signup.html dan tekan tombol enter
3. Ketik perintah seperti di bawah ini:

```

{% extends 'auth/base.html' %}
{% block title %}
    Sing Up
{% endblock %}

{% block body %}
    <div class="container">
        <div class="card o-hidden border-0 shadow-lg
my-5">
            <div class="card-body p-0">
                <!-- Nested Row within Card Body -->
                <div class="row">
                    <div class="col-lg-5 d-none d-lg-block
bg-register-image"></div>
                    <div class="col-lg-7">
                        <div class="p-5">
                            <div class="text-center">
                                <h1 class="h4 text-gray-900 mb-
4">Create an Account!</h1>
                            </div>

```



```

        <form class="user" action="{% url
'signup_admin' %}" method="post">
            {% csrf_token %}
            <div class="form-group">
                <input type="text" class="form-
control form-control-user" id="exampleFirstName"
placeholder="Name" name="name" />
            </div>
            <div class="form-group">
                <input type="email" class="form-
control form-control-user" id="exampleInputEmail"
placeholder="Email Address" name="email" />
            </div>
            <div class="form-group row">
                <div class="col-sm-6 mb-3 mb-sm-
0">
                    <input type="password"
class="form-control form-control-user"
id="exampleInputPassword" placeholder="Password"
name="password" />
                </div>
                <div class="col-sm-6">
                    <input type="password"
class="form-control form-control-user"
id="exampleRepeatPassword" placeholder="Repeat
Password" name="repeat_password" />
                </div>
            </div>
            <input type="submit" class="btn
btn-primary btn-user btn-block" value="Register
Account">
        </form>
        <hr />
        <div class="text-center">

```



```

    <meta http-equiv="X-UA-Compatible"
content="IE=edge">
    <meta name="viewport" content="width=device-
width, initial-scale=1, shrink-to-fit=no">
    <meta name="description" content="">
    <meta name="author" content="">

<title>SB Admin 2 - Dashboard</title>

<!-- Custom fonts for this template-->
<link href="{% static 'vendor/fontawesome-
free/css/all.min.css' %}" rel="stylesheet"
type="text/css">
<link
href="https://fonts.googleapis.com/css?family=Nuni
to:200,200i,300,300i,400,400i,600,600i,700,700i,80
0,800i,900,900i"
rel="stylesheet">
<!-- Custom styles for this template-->
<link href="{% static 'css/sb-admin-2.min.css'
%}" rel="stylesheet">

</head>

<body id="page-top">

    {% block body %}{% endblock %}

<!-- Scroll to Top Button-->
<a class="scroll-to-top rounded" href="#page-
top">
    <i class="fas fa-angle-up"></i>

```

```

</a>

<!-- Logout Modal-->
<div class="modal fade" id="logoutModal"
tabindex="-1" role="dialog" aria-
labelledby="exampleModalLabel"
aria-hidden="true">
  <div class="modal-dialog" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title"
id="exampleModalLabel">Ready to Leave?</h5>
        <button class="close"
type="button" data-dismiss="modal" aria-
label="Close">
          <span aria-
hidden="true">×</span>
        </button>
      </div>
      <div class="modal-body">Select
"Logout" below if you are ready to end your
current session.</div>
      <div class="modal-footer">
        <button class="btn btn-
secondary" type="button" data-
dismiss="modal">Cancel</button>
        <a class="btn btn-primary"
href="login.html">Logout</a>
      </div>
    </div>
  </div>
</div>
<!-- Bootstrap core JavaScript-->

```

```

    <script src="{% static
'vendor/jquery/jquery.min.js' %}" "></script>
    <script src="{% static
'vendor/bootstrap/js/bootstrap.bundle.min.js' %}"
"></script>

    <!-- Core plugin JavaScript-->
    <script src="{% static 'vendor/jquery-
easing/jquery.easing.min.js' %}" "></script>

    <!-- Custom scripts for all pages-->
    <script src="{% static 'js/sb-admin-2.min.js'
%}" "></script>

    <!-- Page level plugins -->
    <script src="{% static
'vendor/chart.js/Chart.min.js' %}" "></script>

    <!-- Page level custom scripts -->
    <script src="{% static 'js/demo/chart-area-
demo.js' %}" "></script>
    <script src="{% static 'js/demo/chart-pie-
demo.js' %}" "></script>

    {% comment %} ckeditor {% endcomment %}
    <script src="{% static 'ckeditor/ckeditor-
init.js' %}" "></script>
    <script src="{% static
'ckeditor/ckeditor/ckeditor.js' %}" "></script>

</body>

</html>

```

## Membuat file sidebar.html di dalam folder templates/blog\_admin di app admin\_blog

Langkah-langkah membuat file sidebar.html di dalam folder templates/blog\_admin di app admin\_blog sebagai berikut:

1. Klik kanan folder blog di bawah folder templates dan pilih menu new file
2. Ketik sidebar.html dan tekan tombol enter
3. Ketik perintah seperti di bawah ini:

```
<!-- Sidebar -->
<ul class="navbar-nav bg-gradient-primary sidebar sidebar-dark accordion" id="accordionSidebar">
  <!-- Sidebar - Brand -->
  <a class="sidebar-brand d-flex align-items-center justify-content-center" href="index.html">
    <div class="sidebar-brand-icon rotate-n-15">
      <i class="fas fa-laugh-wink"></i>
    </div>
    <div class="sidebar-brand-text mx-3">
      SB Admin <sup>2</sup>
    </div>
  </a>

  <!-- Divider -->
  <hr class="sidebar-divider my-0" />

  <!-- Nav Item - Dashboard -->
  <li class="nav-item active">
    <a class="nav-link" href="index.html">
      <i class="fas fa-fw fa-tachometer-alt"></i>
      <span>Dashboard</span>
    </a>
  </li>

  <!-- Divider -->
```

```

<hr class="sidebar-divider" />
<!-- Heading -->
<div class="sidebar-heading">Interface</div>
<!-- Nav Item - Pages Collapse Menu -->
<li class="nav-item">
    <a class="nav-link collapsed" href="#" data-
toggle="collapse" data-target="#collapseTwo" aria-
expanded="true" aria-controls="collapseTwo">
        <i class="fas fa-fw fa-cog"></i>
        <span>Blog</span>
    </a>
    <div id="collapseTwo" class="collapse" aria-
labelledby="headingTwo" data-
parent="#accordionSidebar">
        <div class="bg-white py-2 collapse-inner
rounded">
            <h6 class="collapse-header">Blog
Management:</h6>
            <a class="collapse-item" href="{% url
'artikel_list_admin' %}">List Artikel</a>
            <a class="collapse-item" href="{% url
'artikel_add_admin' %}">Add Artikel</a>
            <a class="collapse-item" href="{% url
'category_list_admin' %}">Category</a>
        </div>
    </div>
</li>

<!-- Nav Item - Utilities Collapse Menu -->
<li class="nav-item">
    <a class="nav-link collapsed" href="#" data-
toggle="collapse" data-target="#collapseUtilities"
aria-expanded="true" aria-
controls="collapseUtilities">

```

```

        <i class="fas fa-fw fa-wrench"></i>
        <span>Admin Blog</span>
    </a>
    <div id="collapseUtilities" class="collapse"
aria-labelledby="headingUtilities" data-
parent="#accordionSidebar">
        <div class="bg-white py-2 collapse-inner
rounded">
            <h6 class="collapse-header">Admin
Management:</h6>
            <a class="collapse-item" href="">List
Admin</a>
            <a class="collapse-item" href="utilities-
border.html">Add Admin</a>
        </div>
    </div>
</li>
<!-- Nav Item - Utilities Collapse Menu -->
<li class="nav-item">
    <a class="nav-link collapsed" href="#">
        <i class="fas fa-envelope-open-text"></i>
        <span>Inbox</span>
    </a>
</li>

<!-- Divider -->
<hr class="sidebar-divider" />

<!-- Heading -->
<div class="sidebar-heading">More</div>
<!-- Nav Item - Tables -->
<li class="nav-item">
    <a class="nav-link" href="">
        <i class="fas fa-fw fa-table"></i>

```



```

        <span>Logout</span>
    </a>
</li>
<!-- Divider -->
<hr class="sidebar-divider d-none d-md-block" />
<!-- Sidebar Toggler (Sidebar) -->
<div class="text-center d-none d-md-inline">
    <button class="rounded-circle border-0"
id="sidebarToggle"></button>
</div>
</ul>
<!-- End of Sidebar -->

```

## Membuat file `artikel_add.html` di dalam folder `templates/blog_admin` di app `admin_blog`

Langkah-langkah membuat file `artikel_add.html` di dalam folder `templates/blog_admin` di app `admin_blog` sebagai berikut:

1. Klik kanan folder `blog_admin` di bawah folder `templates` dan pilih menu `new file`
2. Ketik `artikel_add.html` dan tekan tombol `enter`
3. Ketik perintah seperti di bawah ini:

```

{% extends 'blog_admin/base.html' %}

{% block body %}
    <!-- Page Wrapper -->
    <div id="wrapper">
        {% include 'blog_admin/sidebar.html' %}

        <!-- Content Wrapper -->
        <div id="content-wrapper" class="d-flex flex-
column">
            <!-- Main Content -->
            <div id="content">
                <!-- Topbar -->

```

```

    <nav class="navbar navbar-expand navbar-
light bg-white topbar mb-4 static-top shadow">
    <!-- Sidebar Toggle (Topbar) -->
    <form class="form-inline">
        <button id="sidebarToggleTop"
class="btn btn-link d-md-none rounded-circle mr-
3"><i class="fa fa-bars"></i></button>
    </form>

    <!-- Topbar Search -->
    <form class="d-none d-sm-inline-block
form-inline mr-auto ml-md-3 my-2 my-md-0 mw-100
navbar-search">
        <div class="input-group">
            <input type="text" class="form-
control bg-light border-0 small"
placeholder="Search for..." aria-label="Search"
aria-describedby="basic-addon2" />
            <div class="input-group-append">
                <button class="btn btn-primary"
type="button"><i class="fas fa-search fa-
sm"></i></button>
            </div>
        </div>
    </form>

    <!-- Topbar Navbar -->
    <ul class="navbar-nav ml-auto">
        <!-- Nav Item - Search Dropdown
(Visible Only XS) -->
        <li class="nav-item dropdown no-arrow
d-sm-none">
            <a class="nav-link dropdown-toggle"
href="#" id="searchDropdown" role="button" data-

```

```

toggle="dropdown" aria-haspopup="true" aria-
expanded="false"><i class="fas fa-search fa-
fw"></i></a>
        <!-- Dropdown - Messages -->
        <div class="dropdown-menu dropdown-
menu-right p-3 shadow animated--grow-in" aria-
labelledby="searchDropdown">
            <form class="form-inline mr-auto
w-100 navbar-search">
                <div class="input-group">
                    <input type="text"
class="form-control bg-light border-0 small"
placeholder="Search for..." aria-label="Search"
aria-describedby="basic-addon2" />
                    <div class="input-group-
append">
                        <button class="btn btn-
primary" type="button"><i class="fas fa-search fa-
sm"></i></button>
                    </div>
                </div>
            </form>
        </div>
    </li>

    <!-- Nav Item - Alerts -->
    <li class="nav-item dropdown no-arrow
mx-1">
        <a class="nav-link dropdown-toggle"
href="#" id="alertsDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
            <i class="fas fa-bell fa-fw"></i>
            <!-- Counter - Alerts -->

```

```

        <span class="badge badge-danger
badge-counter">3</span>
    </a>
    <!-- Dropdown - Alerts -->
    <div class="dropdown-list dropdown-
menu dropdown-menu-right shadow animated--grow-in"
aria-labelledby="alertsDropdown">
        <h6 class="dropdown-header">Alerts
Center</h6>
        <a class="dropdown-item d-flex
align-items-center" href="#">
            <div class="mr-3">
                <div class="icon-circle bg-
primary">
                    <i class="fas fa-file-alt
text-white"></i>
                </div>
            </div>
            <div>
                <div class="small text-gray-
500">December 12, 2019</div>
                <span class="font-weight-
bold">A new monthly report is ready to
download!</span>
            </div>
        </a>
        <a class="dropdown-item d-flex
align-items-center" href="#">
            <div class="mr-3">
                <div class="icon-circle bg-
success">
                    <i class="fas fa-donate
text-white"></i>
                </div>
            </div>

```

```

        </div>
        <div>
            <div class="small text-gray-
500">December 7, 2019</div>$290.29 has been
deposited into your account!
        </div>
    </a>
    <a class="dropdown-item d-flex
align-items-center" href="#">
        <div class="mr-3">
            <div class="icon-circle bg-
warning">
                <i class="fas fa-
exclamation-triangle text-white"></i>
            </div>
        </div>
        <div>
            <div class="small text-gray-
500">December 2, 2019</div>Spending Alert: We've
noticed unusually high spending for your account.
        </div>
    </a>
    <a class="dropdown-item text-
center small text-gray-500" href="#">Show All
Alerts</a>
    </div>
</li>

<!-- Nav Item - Messages -->
<li class="nav-item dropdown no-arrow
mx-1">
    <a class="nav-link dropdown-toggle"
href="#" id="messagesDropdown" role="button" data-

```

```

toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
    <i class="fas fa-envelope fa-
fw"></i>
    <!-- Counter - Messages -->
    <span class="badge badge-danger
badge-counter">7</span>
    </a>
    <!-- Dropdown - Messages -->
    <div class="dropdown-list dropdown-
menu dropdown-menu-right shadow animated--grow-in"
aria-labelledby="messagesDropdown">
        <h6 class="dropdown-
header">Message Center</h6>
        <a class="dropdown-item d-flex
align-items-center" href="#">
            <div class="dropdown-list-image
mr-3">
                
                <div class="status-indicator
bg-success"></div>
            </div>
            <div class="font-weight-bold">
                <div class="text-truncate">Hi
there! I am wondering if you can help me with a
problem I've been having.</div>
                <div class="small text-gray-
500">Emily Fowler · 58m</div>
            </div>
        </a>
        <a class="dropdown-item d-flex
align-items-center" href="#">

```

```

        <div class="dropdown-list-image
mr-3">
            
            <div class="status-
indicator"></div>
        </div>
        <div>
            <div class="text-truncate">I
have the photos that you ordered last month, how
would you like them sent to you?</div>
            <div class="small text-gray-
500">Jae Chun · 1d</div>
        </div>
    </a>
    <a class="dropdown-item d-flex
align-items-center" href="#">
        <div class="dropdown-list-image
mr-3">
            
            <div class="status-indicator
bg-warning"></div>
        </div>
        <div>
            <div class="text-
truncate">Last month's report looks great, I am
very happy with the progress so far, keep up the
good work!</div>
            <div class="small text-gray-
500">Morgan Alvarez · 2d</div>
        </div>
    </a>

```

```

        <a class="dropdown-item d-flex
align-items-center" href="#">
            <div class="dropdown-list-image
mr-3">
                
                <div class="status-indicator
bg-success"></div>
            </div>
            <div>
                <div class="text-truncate">Am
I a good boy? The reason I ask is because someone
told me that people say this to all dogs, even if
they aren't good...</div>
                <div class="small text-gray-
500">Chicken the Dog · 2w</div>
            </div>
        </a>
        <a class="dropdown-item text-
center small text-gray-500" href="#">Read More
Messages</a>
    </div>
</li>

    <div class="topbar-divider d-none d-
sm-block"></div>

    <!-- Nav Item - User Information -->
    <li class="nav-item dropdown no-
arrow">
        <a class="nav-link dropdown-toggle"
href="#" id="userDropdown" role="button" data-

```



```

toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
    <span class="mr-2 d-none d-lg-
inline text-gray-600 small">Douglas McGee</span>
    
    </a>
    <!-- Dropdown - User Information -->
    <div class="dropdown-menu dropdown-
menu-right shadow animated--grow-in" aria-
labelledby="userDropdown">
        <a class="dropdown-item" href="#">
            <i class="fas fa-user fa-sm fa-
fw mr-2 text-gray-400"></i>
            Profile
        </a>
        <a class="dropdown-item" href="#">
            <i class="fas fa-cogs fa-sm fa-
fw mr-2 text-gray-400"></i>
            Settings
        </a>
        <a class="dropdown-item" href="#">
            <i class="fas fa-list fa-sm fa-
fw mr-2 text-gray-400"></i>
            Activity Log
        </a>
        <div class="dropdown-
divider"></div>
        <a class="dropdown-item" href="#"
data-toggle="modal" data-target="#logoutModal">
            <i class="fas fa-sign-out-alt
fa-sm fa-fw mr-2 text-gray-400"></i>
            Logout
        </a>

```

```

        </div>
    </li>
</ul>
</nav>
<!-- End of Topbar -->

{% load widget_tweaks %}
<!-- Begin Page Content -->
<div class="container-fluid">
    <form action="{% url 'artikel_add_admin'
%}" method="post" enctype="multipart/form-data">
        {% csrf_token %}
        <div class="card shadow mb-4">
            <div class="card-header py-3">
                <h6 class="m-0 font-weight-bold
text-primary">Title</h6>
            </div>
            <div class="card-body">
                {{ form.title|add_class:'form-
control' }}
            </div>
        </div>
    </div>
    <div class="row">
        <div class="col-lg-8">
            <!-- Basic Card Example -->
            <div class="card shadow mb-4">
                <div class="card-header py-3">
                    <h6 class="m-0 font-weight-
bold text-primary">Content</h6>
                </div>
                <div class="card-body">{{
form.content|add_class:'form-control' }}</div>
            </div>
        </div>
    </div>

```

```

        <div class="col-lg-4">
            <div class="card shadow mb-4">
                <div class="card-header py-3">
                    <h6 class="m-0 font-weight-
bold text-primary">More</h6>
                </div>
                <div class="mb-3">
                    <label class="form-label mx-
3">Category</label>
                    <div class="card-body">{{
form.category|add_class:'form-control' }}</div>
                </div>
                {% comment %} <div class="mb-3">
                    <label class="form-label mx-
3">admin</label>
                    <div class="card-body">{{
form.admin|add_class:'form-control' }}</div>
                </div> {% endcomment %}
                <div class="mb-3">
                    <label class="form-label mx-
3">Thumbnail</label>
                    <div class="card-body">{{
form.thumbnail }}</div>
                </div>
                <input class="btn btn-primary
btn-icon-split mb-3 mx-5 py-2" type="submit"
value="Publish">

            </div>

        </div>
    </div>
</form>

```

```

        <!-- Page Heading -->

        <!-- /.container-fluid -->
    </div>
    <!-- End of Main Content -->

    <!-- Footer -->
    <footer class="sticky-footer bg-white">
        <div class="container my-auto">
            <div class="copyright text-center my-
auto">
                <span>Copyright &copy; Your Website
2020</span>
            </div>
        </div>
    </footer>
    <!-- End of Footer -->
</div>
<!-- End of Content Wrapper -->
</div>
<!-- End of Page Wrapper -->
{% endblock %}

```

## Membuat file `artikel_list.html` di dalam folder `templates/blog_admin` di `app admin_blog`

Langkah-langkah membuat file `artikel_list.html` di dalam folder `templates/blog_admin` di `app admin_blog` sebagai berikut:

1. Klik kanan folder `blog_admin` di bawah folder `templates` dan pilih menu `new file`
2. Ketik `artikel_add.html` dan tekan tombol `enter`
3. Ketik perintah seperti di bawah ini:

```

{% extends 'blog_admin/base.html' %}

{% block body %}
    <!-- Page Wrapper -->
    <div id="wrapper">

        {% include 'blog_admin/sidebar.html' %}

        <!-- Content Wrapper -->
        <div id="content-wrapper" class="d-flex flex-
column">

            <!-- Main Content -->
            <div id="content">

                <!-- Topbar -->
                <nav class="navbar navbar-expand navbar-
light bg-white topbar mb-4 static-top shadow">

                    <!-- Sidebar Toggle (Topbar) -->
                    <form class="form-inline">
                        <button id="sidebarToggleTop"
class="btn btn-link d-md-none rounded-circle mr-3">
                            <i class="fa fa-bars"></i>
                        </button>
                    </form>

                    <!-- Topbar Search -->
                    <form
                        class="d-none d-sm-inline-block
form-inline mr-auto ml-md-3 my-2 my-md-0 mw-100
navbar-search">
                        <div class="input-group">

```

```

        <input type="text"
class="form-control bg-light border-0 small"
placeholder="Search for..."
        aria-label="Search" aria-
describedby="basic-addon2">
        <div class="input-group-
append">
            <button class="btn btn-
primary" type="button">
                <i class="fas fa-
search fa-sm"></i>
            </button>
        </div>
    </div>
</form>

<!-- Topbar Navbar -->
<ul class="navbar-nav ml-auto">

    <!-- Nav Item - Search Dropdown
(Visible Only XS) -->
    <li class="nav-item dropdown no-
arrow d-sm-none">
        <a class="nav-link dropdown-
toggle" href="#" id="searchDropdown" role="button"
        data-toggle="dropdown"
        aria-haspopup="true" aria-expanded="false">
            <i class="fas fa-search
fa-fw"></i>
        </a>
        <!-- Dropdown - Messages -->
        <div class="dropdown-menu
dropdown-menu-right p-3 shadow animated--grow-in"

```

```

        aria-
labelledby="searchDropdown">
        <form class="form-inline
mr-auto w-100 navbar-search">
            <div class="input-
group">
                <input
type="text" class="form-control bg-light border-0
small"
placeholder="Search for..." aria-label="Search"
                aria-
describedby="basic-addon2">
                    <div
class="input-group-append">
                        <button
class="btn btn-primary" type="button">
                            <i
class="fas fa-search fa-sm"></i>
                        </button>
                    </div>
                </div>
            </form>
        </div>
</li>

<!-- Nav Item - Alerts -->
<li class="nav-item dropdown no-
arrow mx-1">
    <a class="nav-link dropdown-
toggle" href="#" id="alertsDropdown" role="button"
        data-toggle="dropdown"
aria-haspopup="true" aria-expanded="false">

```

```

fw"></i>
        <i class="fas fa-bell fa-
        <!-- Counter - Alerts -->
        <span class="badge badge-
danger badge-counter">3+</span>
        </a>
        <!-- Dropdown - Alerts -->
        <div class="dropdown-list
dropdown-menu dropdown-menu-right shadow animated--
grow-in"
                aria-
labelledby="alertsDropdown">
                <h6 class="dropdown-
header">
                        Alerts Center
                </h6>
                <a class="dropdown-item
d-flex align-items-center" href="#">
                        <div class="mr-3">
                                <div class="icon-
circle bg-primary">
                                        <i class="fas
fa-file-alt text-white"></i>
                                </div>
                                </div>
                                <div>
                                        <div class="small
text-gray-500">December 12, 2019</div>
                                        <span
class="font-weight-bold">A new monthly report is
ready to download!</span>
                                </div>
                                </a>

```



```

        <a class="dropdown-item
d-flex align-items-center" href="#">
            <div class="mr-3">
                <div class="icon-
circle bg-success">
                    <i class="fas
fa-donate text-white"></i>
                </div>
            </div>
            <div>
                <div class="small
text-gray-500">December 7, 2019</div>
                $290.29 has been
deposited into your account!
            </div>
        </a>
        <a class="dropdown-item
d-flex align-items-center" href="#">
            <div class="mr-3">
                <div class="icon-
circle bg-warning">
                    <i class="fas
fa-exclamation-triangle text-white"></i>
                </div>
            </div>
            <div>
                <div class="small
text-gray-500">December 2, 2019</div>
                Pending Alert:
                We've noticed unusually high spending for your
                account.
            </div>
        </a>

```

```

                <a class="dropdown-item
text-center small text-gray-500" href="#">Show All
Alerts</a>
            </div>
        </li>

        <!-- Nav Item - Messages -->
        <li class="nav-item dropdown no-
arrow mx-1">
            <a class="nav-link dropdown-
toggle" href="#" id="messagesDropdown" role="button"
data-toggle="dropdown"
aria-haspopup="true" aria-expanded="false">
                <i class="fas fa-envelope
fa-fw"></i>
                <!-- Counter - Messages -
->
                <span class="badge badge-
danger badge-counter">7</span>
            </a>
            <!-- Dropdown - Messages -->
            <div class="dropdown-list
dropdown-menu dropdown-menu-right shadow animated--
grow-in"
                aria-
labelledby="messagesDropdown">
                <h6 class="dropdown-
header">
                    Message Center
                </h6>
                <a class="dropdown-item
d-flex align-items-center" href="#">
                    <div class="dropdown-
list-image mr-3">

```

```

        
        <div
class="status-indicator bg-success"></div>
        </div>
        <div class="font-
weight-bold">
                <div class="text-
truncate">Hi there! I am wondering if you can help me
with a
                problem I've
been having.</div>
                <div class="small
text-gray-500">Emily Fowler · 58m</div>
                </div>
        </a>
        <a class="dropdown-item
d-flex align-items-center" href="#">
                <div class="dropdown-
list-image mr-3">
                        
                        <div
class="status-indicator"></div>
                </div>
                <div>
                        <div class="text-
truncate">I have the photos that you ordered last
month, how
                        would you
like them sent to you?</div>

```

```

                <div class="small
text-gray-500">Jae Chun · 1d</div>
                </div>
            </a>
            <a class="dropdown-item
d-flex align-items-center" href="#">
                <div class="dropdown-
list-image mr-3">
                    
                    <div
class="status-indicator bg-warning"></div>
                </div>
                <div>
                    <div class="text-
truncate">Last month's report looks great, I am very
happy with the progress so far, keep up the good
work!</div>
                </div>
                <div class="small
text-gray-500">Morgan Alvarez · 2d</div>
                </div>
            </a>
            <a class="dropdown-item
d-flex align-items-center" href="#">
                <div class="dropdown-
list-image mr-3">
                    
                    <div
class="status-indicator bg-success"></div>
                </div>

```

```

                <div>
                    <div class="text-
truncate">Am I a good boy? The reason I ask is
because someone
                                told me that
people say this to all dogs, even if they aren't
good...</div>
                                <div class="small
text-gray-500">Chicken the Dog · 2w</div>
                                </div>
                            </a>
                            <a class="dropdown-item
text-center small text-gray-500" href="#">Read More
Messages</a>
                                </div>
                            </li>

                <div class="topbar-divider d-none
d-sm-block"></div>

                <!-- Nav Item - User Information
-->
                <li class="nav-item dropdown no-
arrow">
                    <a class="nav-link dropdown-
toggle" href="#" id="userDropdown" role="button"
                        data-toggle="dropdown"
                        aria-haspopup="true" aria-expanded="false">
                        <span class="mr-2 d-none
d-lg-inline text-gray-600 small">Douglas McGee</span>
                        

```

```

        </a>
        <!-- Dropdown - User
Information -->
        <div class="dropdown-menu
dropdown-menu-right shadow animated--grow-in"
        aria-
labelledby="userDropdown">
        <a class="dropdown-item"
href="#">
                <i class="fas fa-user
fa-sm fa-fw mr-2 text-gray-400"></i>
                Profile
        </a>
        <a class="dropdown-item"
href="#">
                <i class="fas fa-cogs
fa-sm fa-fw mr-2 text-gray-400"></i>
                Settings
        </a>
        <a class="dropdown-item"
href="#">
                <i class="fas fa-list
fa-sm fa-fw mr-2 text-gray-400"></i>
                Activity Log
        </a>
        <div class="dropdown-
divider"></div>
        <a class="dropdown-item"
href="#" data-toggle="modal" data-
target="#logoutModal">
                <i class="fas fa-
sign-out-alt fa-sm fa-fw mr-2 text-gray-400"></i>
                Logout
        </a>

```

```

        </div>
    </li>

</ul>

</nav>
<!-- End of Topbar -->

<!-- Begin Page Content -->
<div class="container-fluid">

    <!-- Page Heading -->
    <h1 class="h3 mb-2 text-gray-
800">Tables</h1>
    <p class="mb-4">DataTables is a third
party plugin that is used to generate the demo table
below.

    For more information about
DataTables, please visit the <a target="_blank"
href="https://datatables.net">official DataTables
documentation</a>.</p>

    <!-- DataTales Example -->
    <div class="card shadow mb-4">
        <div class="card-header py-3">
            <h6 class="m-0 font-weight-
bold text-primary">DataTables Example</h6>
        </div>
        <div class="card-body">
            <div class="table-
responsive">

```

```

<table class="table
table-bordered" id="dataTable" width="100%"
cellspacing="0">
    <thead>
        <tr>
            <th>Nomor</th>
            <th>Judul</th>
            <th>Kategori</th>
            <th>Admin</th>
            <th>Dibuat</th>
            <th>Diperbarui</th>
            <th>Aksi</th>
        </tr>
    </thead>
    <tbody>
        {% for artikel in
artikel_all %}
        <tr>
            <td>{{forloop.counter}}</td>
            <td>{{artikel.title}}</td>
            <td>{{artikel.category}}</td>
            <td>{{artikel.admin}}</td>
            <td>{{artikel.created_at}}</td>

```



```

<td>{{artikel.updated_at}}</td>
                                <td><a
href="{% url 'artikel_update_admin' artikel.id %}"
class="btn btn-warning btn-circle btn-sm">
                                <i
class="fas fa-exclamation-triangle"></i>
                                </a>
                                <a href="{%
url 'artikel_delete_admin' artikel.id %}" class="btn
btn-danger btn-circle btn-sm">
                                <i
class="fas fa-trash"></i>
                                </a></td>
                                </tr>
                                {% endfor %}
                                </tbody>
                                </table>
                                </div>
                                </div>
                                </div>
                                </div>
                                <!-- /.container-fluid -->

</div>
<!-- End of Main Content -->

<!-- Footer -->
<footer class="sticky-footer bg-white">
    <div class="container my-auto">
        <div class="copyright text-center my-
auto">

```

```

                <span>Copyright &copy; Your
Website 2020</span>
            </div>
        </div>
    </footer>
    <!-- End of Footer -->

</div>
<!-- End of Content Wrapper -->

</div>
<!-- End of Page Wrapper -->
{% endblock %}

```

## Membuat file `artikel_update.html` di dalam folder `templates/blog_admin` di app `admin_blog`

Langkah-langkah membuat file `artikel_update.html` di dalam folder `templates/blog_admin` di app `admin_blog` sebagai berikut:

1. Klik kanan folder `blog_admin` di bawah folder `templates` dan pilih menu `new file`
2. Ketik `artikel_update.html` dan tekan tombol `enter`
3. Ketik perintah seperti di bawah ini:

```

{% extends 'blog_admin/base.html' %}

{% block body %}
    <!-- Page Wrapper -->
    <div id="wrapper">
        {% include 'blog_admin/sidebar.html' %}

        <!-- Content Wrapper -->
        <div id="content-wrapper" class="d-flex flex-
column">
            <!-- Main Content -->

```

```

<div id="content">
  <!-- Topbar -->
  <nav class="navbar navbar-expand navbar-light
bg-white topbar mb-4 static-top shadow">
    <!-- Sidebar Toggle (Topbar) -->
    <form class="form-inline">
      <button id="sidebarToggleTop" class="btn
btn-link d-md-none rounded-circle mr-3"><i class="fa
fa-bars"></i></button>
    </form>

    <!-- Topbar Search -->
    <form class="d-none d-sm-inline-block form-
inline mr-auto ml-md-3 my-2 my-md-0 mw-100 navbar-
search">
      <div class="input-group">
        <input type="text" class="form-control
bg-light border-0 small" placeholder="Search for..."
aria-label="Search" aria-describedby="basic-addon2"
/>

        <div class="input-group-append">
          <button class="btn btn-primary"
type="button"><i class="fas fa-search fa-
sm"></i></button>
        </div>
      </div>
    </form>

    <!-- Topbar Navbar -->
    <ul class="navbar-nav ml-auto">
      <!-- Nav Item - Search Dropdown (Visible
Only XS) -->
      <li class="nav-item dropdown no-arrow d-
sm-none">

```

```

        <a class="nav-link dropdown-toggle"
href="#" id="searchDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-
expanded="false"><i class="fas fa-search fa-
fw"></i></a>
        <!-- Dropdown - Messages -->
        <div class="dropdown-menu dropdown-
menu-right p-3 shadow animated--grow-in" aria-
labelledby="searchDropdown">
            <form class="form-inline mr-auto w-
100 navbar-search">
                <div class="input-group">
                    <input type="text" class="form-
control bg-light border-0 small" placeholder="Search
for..." aria-label="Search" aria-describedby="basic-
addon2" />
                    <div class="input-group-append">
                        <button class="btn btn-primary"
type="button"><i class="fas fa-search fa-
sm"></i></button>
                    </div>
                </div>
            </form>
        </div>
    </li>

    <!-- Nav Item - Alerts -->
    <li class="nav-item dropdown no-arrow mx-
1">
        <a class="nav-link dropdown-toggle"
href="#" id="alertsDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
            <i class="fas fa-bell fa-fw"></i>

```

```

        <!-- Counter - Alerts -->
        <span class="badge badge-danger
badge-counter">3</span>
    </a>
    <!-- Dropdown - Alerts -->
    <div class="dropdown-list dropdown-menu
dropdown-menu-right shadow animated--grow-in" aria-
labelledby="alertsDropdown">
        <h6 class="dropdown-header">Alerts
Center</h6>
        <a class="dropdown-item d-flex align-
items-center" href="#">
            <div class="mr-3">
                <div class="icon-circle bg-
primary">
                    <i class="fas fa-file-alt text-
white"></i>
                </div>
            </div>
            <div>
                <div class="small text-gray-
500">December 12, 2019</div>
                <span class="font-weight-bold">A
new monthly report is ready to download!</span>
            </div>
        </a>
        <a class="dropdown-item d-flex align-
items-center" href="#">
            <div class="mr-3">
                <div class="icon-circle bg-
success">
                    <i class="fas fa-donate text-
white"></i>
                </div>
            </div>

```

```

        </div>
        <div>
            <div class="small text-gray-
500">December 7, 2019</div>$290.29 has been deposited
into your account!
        </div>
    </a>
    <a class="dropdown-item d-flex align-
items-center" href="#">
        <div class="mr-3">
            <div class="icon-circle bg-
warning">
                <i class="fas fa-exclamation-
triangle text-white"></i>
            </div>
        </div>
        <div>
            <div class="small text-gray-
500">December 2, 2019</div>Spending Alert: We've
noticed unusually high spending for your account.
        </div>
    </a>
    <a class="dropdown-item text-center
small text-gray-500" href="#">Show All Alerts</a>
    </div>
</li>

<!-- Nav Item - Messages -->
<li class="nav-item dropdown no-arrow mx-
1">
    <a class="nav-link dropdown-toggle"
href="#" id="messagesDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-
expanded="false">

```

```

        <i class="fas fa-envelope fa-fw"></i>
        <!-- Counter - Messages -->
        <span class="badge badge-danger
badge-counter">7</span>
    </a>
    <!-- Dropdown - Messages -->
    <div class="dropdown-list dropdown-menu
dropdown-menu-right shadow animated--grow-in" aria-
labelledby="messagesDropdown">
        <h6 class="dropdown-header">Message
Center</h6>
        <a class="dropdown-item d-flex align-
items-center" href="#">
            <div class="dropdown-list-image mr-
3">
                
                <div class="status-indicator bg-
success"></div>
            </div>
            <div class="font-weight-bold">
                <div class="text-truncate">Hi
there! I am wondering if you can help me with a
problem I've been having.</div>
                <div class="small text-gray-
500">Emily Fowler · 58m</div>
            </div>
        </a>
        <a class="dropdown-item d-flex align-
items-center" href="#">
            <div class="dropdown-list-image mr-
3">
                

```

```

        <div class="status-
indicator"></div>
        </div>
        <div>
            <div class="text-truncate">I have
the photos that you ordered last month, how would you
like them sent to you?</div>
            <div class="small text-gray-
500">Jae Chun · 1d</div>
            </div>
        </a>
        <a class="dropdown-item d-flex align-
items-center" href="#">
            <div class="dropdown-list-image mr-
3">
                
                <div class="status-indicator bg-
warning"></div>
            </div>
            <div>
                <div class="text-truncate">Last
month's report looks great, I am very happy with the
progress so far, keep up the good work!</div>
                <div class="small text-gray-
500">Morgan Alvarez · 2d</div>
            </div>
        </a>
        <a class="dropdown-item d-flex align-
items-center" href="#">
            <div class="dropdown-list-image mr-
3">

```



```

        
        <div class="status-indicator bg-
success"></div>
    </div>
    <div>
        <div class="text-truncate">Am I a
good boy? The reason I ask is because someone told me
that people say this to all dogs, even if they aren't
good...</div>
        <div class="small text-gray-
500">Chicken the Dog · 2w</div>
    </div>
    </a>
    <a class="dropdown-item text-center
small text-gray-500" href="#">Read More Messages</a>
    </div>
</li>

<div class="topbar-divider d-none d-sm-
block"></div>

<!-- Nav Item - User Information -->
<li class="nav-item dropdown no-arrow">
    <a class="nav-link dropdown-toggle"
href="#" id="userDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
        <span class="mr-2 d-none d-lg-inline
text-gray-600 small">Douglas McGee</span>
        
    </a>

```

```

        <!-- Dropdown - User Information -->
        <div class="dropdown-menu dropdown-
menu-right shadow animated--grow-in" aria-
labelledby="userDropdown">
            <a class="dropdown-item" href="#">
                <i class="fas fa-user fa-sm fa-fw
mr-2 text-gray-400"></i>
                Profile
            </a>
            <a class="dropdown-item" href="#">
                <i class="fas fa-cogs fa-sm fa-fw
mr-2 text-gray-400"></i>
                Settings
            </a>
            <a class="dropdown-item" href="#">
                <i class="fas fa-list fa-sm fa-fw
mr-2 text-gray-400"></i>
                Activity Log
            </a>
            <div class="dropdown-divider"></div>
            <a class="dropdown-item" href="#"
data-toggle="modal" data-target="#logoutModal">
                <i class="fas fa-sign-out-alt fa-sm
fa-fw mr-2 text-gray-400"></i>
                Logout
            </a>
        </div>
    </li>
</ul>
</nav>
<!-- End of Topbar -->

{% load widget_tweaks %}
<!-- Begin Page Content -->

```

```

    <div class="container-fluid">
      <form method="post"
enctype="multipart/form-data">
        {% csrf_token %}
        <div class="card shadow mb-4">
          <div class="card-header py-3">
            <h6 class="m-0 font-weight-bold text-
primary">Title</h6>
          </div>
          <div class="card-body">
            {{ form.title|add_class:'form-
control' }}
          </div>
        </div>
      </div>
      <div class="row">
        <div class="col-lg-8">
          <!-- Basic Card Example -->
          <div class="card shadow mb-4">
            <div class="card-header py-3">
              <h6 class="m-0 font-weight-bold
text-primary">Content</h6>
            </div>
            <div class="card-body">{{
form.content|add_class:'form-control' }}</div>
          </div>
        </div>
        <div class="col-lg-4">
          <div class="card shadow mb-4">
            <div class="card-header py-3">
              <h6 class="m-0 font-weight-bold
text-primary">More</h6>
            </div>
          <div class="mb-3">

```

```

        <label class="form-label mx-
3">Category</label>
        <div class="card-body">{{
form.category|add_class:'form-control' }}</div>
        </div>
        {% comment %} <div class="mb-3">
        <label class="form-label mx-
3">admin</label>
        <div class="card-body">{{
form.admin|add_class:'form-control' }}</div>
        </div> {% endcomment %}
        <div class="mb-3">
        <label class="form-label mx-
3">Thumbnail</label>
        <div class="card-body">{{
form.thumbnail }}</div>
        </div>
        <input class="btn btn-primary btn-
icon-split mb-3 mx-5 py-2" type="submit"
value="Publish">

        </div>

        </div>
</div>

</form>
<!-- Page Heading -->

<!-- /.container-fluid -->
</div>
<!-- End of Main Content -->

<!-- Footer -->

```

```

    <footer class="sticky-footer bg-white">
      <div class="container my-auto">
        <div class="copyright text-center my-auto">
          <span>Copyright &copy; Your Website
2020</span>
        </div>
      </div>
    </footer>
    <!-- End of Footer -->
  </div>
  <!-- End of Content Wrapper -->
</div>
<!-- End of Page Wrapper -->
{% endblock %}

```

## Membuat file `artikel_category_list.html` di dalam folder `templates/blog_admin` di app `admin_blog`

Langkah-langkah membuat file `artikel_category_list.html` di dalam folder `templates/blog_admin` di app `admin_blog` sebagai berikut:

1. Klik kanan folder `blog_admin` di bawah folder `templates` dan pilih menu `new file`
2. Ketik `artikel_category_list.html` dan tekan tombol `enter`
3. Ketik perintah seperti di bawah ini:

```

{% extends 'blog_admin/base.html' %}

{% block body %}
  <!-- Page Wrapper -->
  <div id="wrapper">
    {% include 'blog_admin/sidebar.html' %}

    <!-- Content Wrapper -->
    <div id="content-wrapper" class="d-flex flex-
column">
      <!-- Main Content -->

```

```

<div id="content">
  <!-- Topbar -->
  <nav class="navbar navbar-expand navbar-light
bg-white topbar mb-4 static-top shadow">
    <!-- Sidebar Toggle (Topbar) -->
    <form class="form-inline">
      <button id="sidebarToggleTop" class="btn
btn-link d-md-none rounded-circle mr-3"><i class="fa
fa-bars"></i></button>
    </form>

    <!-- Topbar Search -->
    <form class="d-none d-sm-inline-block form-
inline mr-auto ml-md-3 my-2 my-md-0 mw-100 navbar-
search">
      <div class="input-group">
        <input type="text" class="form-control
bg-light border-0 small" placeholder="Search for..."
aria-label="Search" aria-describedby="basic-addon2"
/>

        <div class="input-group-append">
          <button class="btn btn-primary"
type="button"><i class="fas fa-search fa-
sm"></i></button>
        </div>
      </div>
    </form>

    <!-- Topbar Navbar -->
    <ul class="navbar-nav ml-auto">
      <!-- Nav Item - Search Dropdown (Visible
Only XS) -->
      <li class="nav-item dropdown no-arrow d-
sm-none">

```

```

        <a class="nav-link dropdown-toggle"
href="#" id="searchDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-
expanded="false"><i class="fas fa-search fa-
fw"></i></a>
        <!-- Dropdown - Messages -->
        <div class="dropdown-menu dropdown-
menu-right p-3 shadow animated--grow-in" aria-
labelledby="searchDropdown">
            <form class="form-inline mr-auto w-
100 navbar-search">
                <div class="input-group">
                    <input type="text" class="form-
control bg-light border-0 small" placeholder="Search
for..." aria-label="Search" aria-describedby="basic-
addon2" />
                    <div class="input-group-append">
                        <button class="btn btn-primary"
type="button"><i class="fas fa-search fa-
sm"></i></button>
                    </div>
                </div>
            </form>
        </div>
    </li>

    <!-- Nav Item - Alerts -->
    <li class="nav-item dropdown no-arrow mx-
1">
        <a class="nav-link dropdown-toggle"
href="#" id="alertsDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
            <i class="fas fa-bell fa-fw"></i>

```

```

        <!-- Counter - Alerts -->
        <span class="badge badge-danger
badge-counter">3</span>
    </a>
    <!-- Dropdown - Alerts -->
    <div class="dropdown-list dropdown-menu
dropdown-menu-right shadow animated--grow-in" aria-
labelledby="alertsDropdown">
        <h6 class="dropdown-header">Alerts
Center</h6>
        <a class="dropdown-item d-flex align-
items-center" href="#">
            <div class="mr-3">
                <div class="icon-circle bg-
primary">
                    <i class="fas fa-file-alt text-
white"></i>
                </div>
            </div>
            <div>
                <div class="small text-gray-
500">December 12, 2019</div>
                <span class="font-weight-bold">A
new monthly report is ready to download!</span>
            </div>
        </a>
        <a class="dropdown-item d-flex align-
items-center" href="#">
            <div class="mr-3">
                <div class="icon-circle bg-
success">
                    <i class="fas fa-donate text-
white"></i>
                </div>
            </div>

```



```

        </div>
        <div>
            <div class="small text-gray-
500">December 7, 2019</div>$290.29 has been deposited
into your account!
        </div>
    </a>
    <a class="dropdown-item d-flex align-
items-center" href="#">
        <div class="mr-3">
            <div class="icon-circle bg-
warning">
                <i class="fas fa-exclamation-
triangle text-white"></i>
            </div>
        </div>
        <div>
            <div class="small text-gray-
500">December 2, 2019</div>Spending Alert: We've
noticed unusually high spending for your account.
        </div>
    </a>
    <a class="dropdown-item text-center
small text-gray-500" href="#">Show All Alerts</a>
    </div>
</li>

<!-- Nav Item - Messages -->
<li class="nav-item dropdown no-arrow mx-
1">
    <a class="nav-link dropdown-toggle"
href="#" id="messagesDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-
expanded="false">

```

```

        <i class="fas fa-envelope fa-fw"></i>
        <!-- Counter - Messages -->
        <span class="badge badge-danger
badge-counter">7</span>
    </a>
    <!-- Dropdown - Messages -->
    <div class="dropdown-list dropdown-menu
dropdown-menu-right shadow animated--grow-in" aria-
labelledby="messagesDropdown">
        <h6 class="dropdown-header">Message
Center</h6>
        <a class="dropdown-item d-flex align-
items-center" href="#">
            <div class="dropdown-list-image mr-
3">
                
                <div class="status-indicator bg-
success"></div>
            </div>
            <div class="font-weight-bold">
                <div class="text-truncate">Hi
there! I am wondering if you can help me with a
problem I've been having.</div>
                <div class="small text-gray-
500">Emily Fowler · 58m</div>
            </div>
        </a>
        <a class="dropdown-item d-flex align-
items-center" href="#">
            <div class="dropdown-list-image mr-
3">
                

```

```

        <div class="status-
indicator"></div>
        </div>
        <div>
            <div class="text-truncate">I have
the photos that you ordered last month, how would you
like them sent to you?</div>
            <div class="small text-gray-
500">Jae Chun · 1d</div>
            </div>
        </a>
        <a class="dropdown-item d-flex align-
items-center" href="#">
            <div class="dropdown-list-image mr-
3">
                
                <div class="status-indicator bg-
warning"></div>
            </div>
            <div>
                <div class="text-truncate">Last
month's report looks great, I am very happy with the
progress so far, keep up the good work!</div>
                <div class="small text-gray-
500">Morgan Alvarez · 2d</div>
            </div>
        </a>
        <a class="dropdown-item d-flex align-
items-center" href="#">
            <div class="dropdown-list-image mr-
3">

```

```

        
        <div class="status-indicator bg-
success"></div>
    </div>
    <div>
        <div class="text-truncate">Am I a
good boy? The reason I ask is because someone told me
that people say this to all dogs, even if they aren't
good...</div>
        <div class="small text-gray-
500">Chicken the Dog · 2w</div>
    </div>
    </a>
    <a class="dropdown-item text-center
small text-gray-500" href="#">Read More Messages</a>
    </div>
</li>

    <div class="topbar-divider d-none d-sm-
block"></div>

    <!-- Nav Item - User Information -->
    <li class="nav-item dropdown no-arrow">
        <a class="nav-link dropdown-toggle"
href="#" id="userDropdown" role="button" data-
toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
            <span class="mr-2 d-none d-lg-inline
text-gray-600 small">Douglas McGee</span>
            
        </a>

```

```

        <!-- Dropdown - User Information -->
        <div class="dropdown-menu dropdown-
menu-right shadow animated--grow-in" aria-
labelledby="userDropdown">
            <a class="dropdown-item" href="#">
                <i class="fas fa-user fa-sm fa-fw
mr-2 text-gray-400"></i>
                Profile
            </a>
            <a class="dropdown-item" href="#">
                <i class="fas fa-cogs fa-sm fa-fw
mr-2 text-gray-400"></i>
                Settings
            </a>
            <a class="dropdown-item" href="#">
                <i class="fas fa-list fa-sm fa-fw
mr-2 text-gray-400"></i>
                Activity Log
            </a>
            <div class="dropdown-divider"></div>
            <a class="dropdown-item" href="#"
data-toggle="modal" data-target="#logoutModal">
                <i class="fas fa-sign-out-alt fa-sm
fa-fw mr-2 text-gray-400"></i>
                Logout
            </a>
        </div>
    </li>
</ul>
</nav>
<!-- End of Topbar -->

<!-- Begin Page Content -->
<div class="container-fluid">

```

```

        <!-- Page Heading -->
        <h1 class="h3 mb-2 text-gray-
800">category</h1>
        <form action="{% url 'category_add_admin'
%}" method="post">
            {% csrf_token %}
            <div class="row mb-3">
                <label for="inputEmail3" class="col-sm-
2 col-form-label">Category name</label>
                <div class="col-md-6">
                    <input type="text" class="form-
control" id="category_name" name="title" />
                </div>
                <input type="submit" class="btn btn-
primary mb-2" value="Tambah Category" />
            </div>
        </form>

        <!-- DataTales Example -->
        <div class="card shadow mb-4">
            <div class="card-header py-3">
                <h6 class="m-0 font-weight-bold text-
primary">Category List</h6>
            </div>
            <div class="card-body">
                <div class="table-responsive">
                    <table class="table table-bordered"
id="dataTable" width="100%" cellpadding="0">
                        <thead>
                            <tr>
                                <th>ID</th>
                                <th>Title</th>
                                <th>Created At</th>
                                <th>Aksi</th>

```

```

        </tr>
    </thead>
    <tbody>

        {% for artikel in artikel_all %}
        <tr>
            <td>{{ artikel.id }}</td>
            <td>{{ artikel.title }}</td>
            <td>{{ artikel.created_at
}}</td>
            <td>
                <a href="#" class="btn
btn-warning btn-circle btn-sm">
                    <i class="fas fa-
exclamation-triangle"></i>
                </a>
                <a href="#" class="btn
btn-danger btn-circle btn-sm">
                    <i class="fas fa-
trash"></i>
                </a>
            </td>
        </tr>
        {% endfor %}

    </tbody>
</table>
</div>
</div>
</div>
</div>
<!-- /.container-fluid -->
</div>

```

```

<!-- End of Main Content -->

<!-- Footer -->
<footer class="sticky-footer bg-white">
  <div class="container my-auto">
    <div class="copyright text-center my-auto">
      <span>Copyright &copy; Your Website
2020</span>
    </div>
  </div>
</footer>
<!-- End of Footer -->
</div>
<!-- End of Content Wrapper -->
</div>
<!-- End of Page Wrapper -->
{% endblock %}

```

## Memodifikasi file models.py di app admin\_blog

Langkah Langkag memodifikasi file models.py di dalam folder blog sebagai berikut:

1. Klik file models.py di dalam folder admin\_blog
2. Ubah kode menjadi seperti berikut

```

# Create your models here.
from django.db import models
from django.urls import reverse

```

```

# Create your models here.

```

```

class admin(models.Model):
    id = models.AutoField(primary_key= True)
    name = models.CharField(max_length=50)
    email = models.EmailField(max_length=254)

```



```

password = models.CharField(max_length=50)
created_at =
models.DateTimeField(auto_now_add=True)
updated_at = models.DateTimeField(auto_now=True)

class Meta:
    verbose_name = ("admin")
    verbose_name_plural = ("admins")
    db_table = 'admin_blog'

def __str__(self):
    return self.name

def get_absolute_url(self):
    return reverse("admin_detail", kwargs={"pk":
self.pk})

```

## Menambahkan file forms.py di dalam app admin\_blog

Langkah langkah menambah file forms.py di dalam folder blog, sebagai berikut:

1. Klik kanan di folder blog dan pilih new file
2. Ketik forms.py dan tekan tombol enter
3. Masukkan kode sebagai berikut

```

from django import forms
from blog.models import blog
from ckeditor.fields import CKEditorWidget

```

```

class ArtikelForm(forms.ModelForm):
    class Meta:
        model = blog
        fields = ['title', 'content', 'thumbnail',
'admin', 'category']
        widgets = {

```

```
        'content': CKEditorWidget(),
    }
```

## Memodifikasi file views.py di app admin\_blog

Langkah Langkah memodifikasi file views.py di dalam folder blog sebagai berikut:

1. Klik file views.py di dalam folder admin\_blog
2. Ubah kode menjadi seperti berikut:

```
from django.shortcuts import render, redirect,
get_object_or_404
from django.http import HttpResponseRedirect
from .models import admin
from blog.models import blog, category
from django.contrib import messages
from .forms import ArtikelForm

# Create your views here.
def signup(request):
    if request.method == 'POST':
        name = request.POST.get('name')
        email = request.POST.get('email')
        password = request.POST.get('password')
        repeat_password =
request.POST.get('repeat_password')
        print(name, email, password, repeat_password)
        if password == repeat_password :
            if
admin.objects.filter(email=email).count()>0:
                messages.error(request, 'Username
already exists.')
            else:
                user = admin(name=name, email=email,
password=password)
                user.save()
```

```

        return redirect('login_admin')
    else:
        messages.error(request, 'Passwords do not
match.')
```

```

    elif request.method == 'GET':
        if 'user' in request.session:
            return redirect('dashboard_admin')
        else:
            return render(request,
'auth/signup.html')
```

```

def login(request):
    if request.method == 'POST':
        email = request.POST.get('email')
        password = request.POST.get('password')

        check_user =
admin.objects.filter(email=email, password=password)
        if check_user:
            request.session['user'] = email
            return redirect('dashboard_admin')
        else:
            messages.error(request, 'Please enter
valid Username or Password.')
```

```

            return redirect('login_admin')
    elif request.method == 'GET':
        if 'user' in request.session:
            return redirect('dashboard_admin')
        else:
            return render(request, 'auth/login.html')
```

```

def logout(request):
    try:
        del request.session['user']
    except:
        return redirect('login_admin')
    return redirect('login_admin')

def dashboard(request):
    if 'user' in request.session:
        current_user = request.session['user']
        param = {'current_user': current_user}
        return
    render(request, "blog_admin/dashboard.html", param)
    else:
        return redirect('login')

def category_list(request):
    if 'user' in request.session:
        context = {
            "title" : "Artikel List",
            "artikel_all" : category.objects.all()
        }
        print(context)
        return
    render(request, "blog_admin/artikel_category_list.html", context)
    else:
        return redirect('login_admin')

def category_add(request):
    if 'user' in request.session:
        if request.method == 'POST':

```

```

        title = request.POST.get('title')
        category(title=title).save()
        messages.error(request, 'Data berhasil
Dihapus')
        return redirect("category_list_admin")
    else :
        messages.error(request, 'Terjadi
kesalahan')
        return redirect("category_list_admin")
    else:
        return redirect('login_admin')
def category_delete(request, id):
    if 'user' in request.session:
        try:
            categoryV = category.objects.get(id=id)
            categoryV.delete()
            return redirect('category_list_admin')
        except :
            return redirect('category_list_admin')
    else:
        return redirect('login_admin')

def artikel_list(request):
    if 'user' in request.session:
        context = {
            "title" : "Artikel List",
            "artikel_all" : blog.objects.all()
        }
        print(context)
        return
    render(request, "blog_admin/artikel_list.html",
context)
    else:

```

```

        return redirect('login_admin')

def artikel_add(request):
    if 'user' in request.session:

        if request.method == 'POST':
            title = request.POST.get('title')
            content = request.POST.get('content')
            categoryS =
category.objects.get(pk=request.POST.get('category'))
            thumbnail = request.FILES['thumbnail']
            admins =
admin.objects.get(email=request.session['user'])
            blog(title=title, content=content,
category=categoryS, thumbnail=thumbnail,
admin=admins).save()
            print("BERHASIL")
            return redirect("artikel_list_admin")
        else :
            return
render(request, "blog_admin/artikel_add.html",
{'form': ArtikelForm()})
    else:
        return redirect('login_admin')

def artikel_update(request, id):
    if 'user' in request.session:
        blogUpdate = get_object_or_404(blog, pk=id)
        if request.method == 'POST':

            blogUpdate.title =
request.POST.get('title')
            blogUpdate.content =
request.POST.get('content')

```

```

        blogUpdate.category =
category.objects.get(pk=request.POST.get('category'))
        if request.FILES["thumbnail"] != "" :
            blogUpdate.thumbnail =
request.FILES['thumbnail']
            blogUpdate.save()

        print("BERHASIL")
        return redirect("artikel_list_admin")
    else :
        return
render(request,"blog_admin/artikel_update.html",
{'form': ArtikelForm(instance=blogUpdate)})
    else:
        return redirect('login_admin')

def artikel_delete(request,id):
    if 'user' in request.session:
        blogdelete = blog.objects.get(pk=id)
        blogdelete.delete()
        return redirect("artikel_list_admin")
    else:
        return redirect('login_admin')

```

## Memodifikasi file urls.py

Langkah- langkah mengatur file seting.py sebagai berikut:

1. pergi ke app admin\_blog lalu klik new file pada di dalam folder admin\_blog lalu buat dengan nama urls.py
2. Lalu tambahkna kode seperti berikut ini

```
from django.urls import path
from . import views
urlpatterns = [
    path('login/', views.login, name='login_admin'),
    path('logout/', views.logout,
name='logout_admin'),
    path('signup/', views.signup,
name='signup_admin'),
    path("", views.dashboard ,
name="dashboard_admin"),
    path("artikel", views.artikel_list ,
name="artikel_list_admin"),
    path("artikel/add", views.artikel_add ,
name="artikel_add_admin"),
    path("artikel/update/<int:id>",
views.artikel_update , name="artikel_update_admin"),
    path("artikel/delete/<int:id>",
views.artikel_delete , name="artikel_delete_admin"),
    path("category", views.category_list ,
name="category_list_admin"),
    path("category/add", views.category_add ,
name="category_add_admin"),
]
```

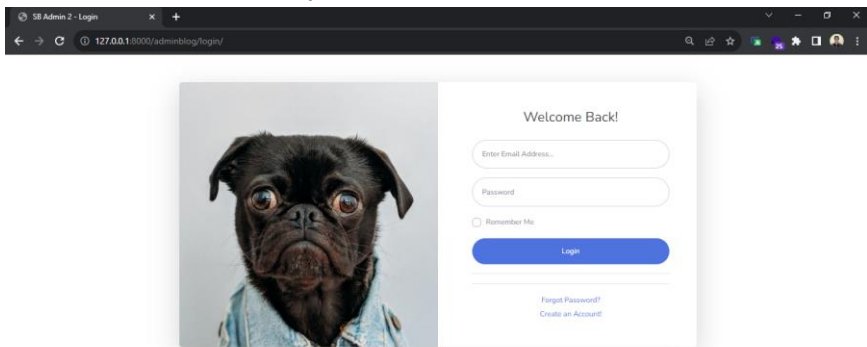
## Melakukan Migration ke project admin blog

Berikut langkah langkah migrasi database di project yang telah kita buat ini

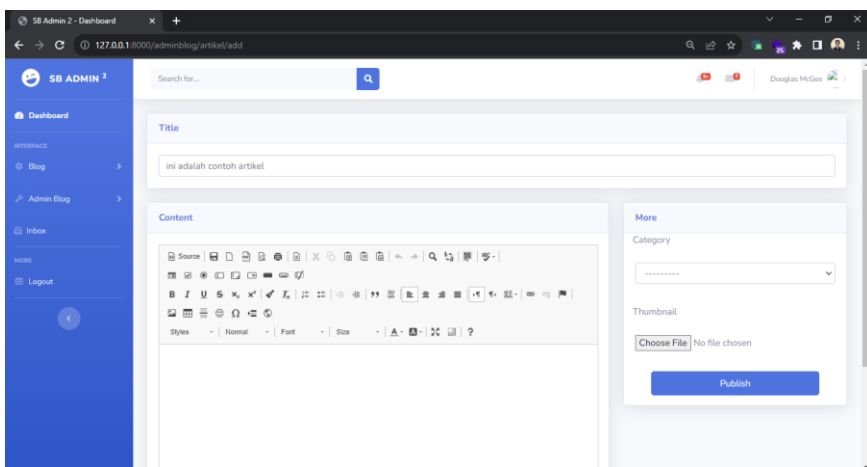


1. Masuk ke terminal visual studi code
2. Buat file migration di project kita dengan perintah berikut ini:  
*python manage.py makemigrations*
3. Import file migration ke database dengan perintah sebagai berikut:  
*python manage.py migrate*

Maka admin blog akan yang kita rancang akan seperti ini:




**Gambar 76. Halaman Admin Blog**



**Gambar 77. Halaman Blog**





## BAB VI

# PENUTUP

**D**alam perjalanan ini, kita telah menjelajahi dunia pemrograman web dengan menggunakan framework Django. Kami berharap buku ini telah memberikan pemahaman yang kuat tentang bagaimana membangun aplikasi web menggunakan Python dan Django. Sebagai pemula, Anda telah mengambil langkah pertama yang penting dalam menguasai keterampilan yang sangat berharga dalam dunia pengembangan web. Django adalah alat yang kuat dan fleksibel yang dapat membantu Anda membangun aplikasi web yang tangguh dan efisien. Namun, ini hanya awal dari perjalanan Anda. Dalam dunia pemrograman, pembelajaran tidak pernah berakhir. Teruslah eksplorasi dan berlatih, dan Anda akan terkejut dengan seberapa jauh Anda bisa pergi.

Selamat atas pencapaian Anda dalam menyelesaikan buku ini. Teruslah belajar, teruslah mengembangkan proyek-proyek Anda sendiri, dan jangan ragu untuk bertanya dan berkolaborasi dengan komunitas pengembang Django yang luas. Semoga buku ini telah memberikan dasar yang kuat bagi Anda untuk melanjutkan perjalanan pemrograman web Anda. Akhirnya, kami ingin mengucapkan terima kasih atas dukungan Anda dalam menjalani perjalanan ini. Semoga Anda sukses dalam semua upaya Anda dalam dunia pemrograman web dengan Django dan bahasa pemrograman.



## DAFTAR PUSTAKA

Adhi Prasetyo.2014. Buku Sakti Webmaster (PHP dan Mysql, HTML dan CSS, HTML5). Jakarta : PT.Transmedia

Erinno Viardi.2012.Pengontrolan Sistem. Yogyakarta : ANDI

Ir. Yuniar Supardi, dkk. 2022.Trik Jitu Belajar Web Python (Django 3.x). Jakarta : PT. Elexmedia Komputindo

Kadir Abdul. 2005. Dasar Pemrograman Python.Yogyakarta : Penerbit Andi